Université de Montréal

Personalized Question-based Cybersecurity Recommendation Systems

Par

Suzy Edith Moukala Both

Département d’Informatique et de Recherche Opérationnelle, Faculté des Arts et des Sciences

Mémoire présenté en vue de l’obtention du grade de Maître ès sciences (M.Sc.)

en informatique

Juillet 2021

Université de Montréal

Département d'Informatique et de Recherche Opérationnelle, Faculté des Arts et des Sciences

Ce mémoire intitulé

Personalized Question-based Cybersecurity

Recommendation Systems

Présenté par

Suzy Edith Moukala Both

A été évalué(e) par un jury composé des personnes suivantes

Claude Frasson

Président-rapporteur

Esma Aïmeur

Directrice de recherche

Yoshua Bengio

Membre du jury

Résumé

En ces temps de pandémie Covid19, une énorme quantité de l’activité humaine est modifiée pour se faire à distance, notamment par des moyens électroniques. Cela rend plusieurs personnes et services vulnérables aux cyberattaques, d’où le besoin d’une éducation généralisée ou du moins accessible sur la cybersécurité. De nombreux efforts sont entrepris par les chercheurs, le gouvernement et les entreprises pour protéger et assurer la sécurité des individus contre les pirates et les cybercriminels. En raison du rôle important joué par les systèmes de recommandation dans la vie quotidienne de l'utilisateur, il est intéressant de voir comment nous pouvons combiner les systèmes de cybersécurité et de recommandation en tant que solutions alternatives pour aider les utilisateurs à comprendre les cyberattaques auxquelles ils peuvent être confrontés. Les systèmes de recommandation sont couramment utilisés par le commerce électronique, les réseaux sociaux et les plateformes de voyage, et ils sont basés sur des techniques de systèmes de recommandation traditionnels.

Au vu des faits mentionnés ci-dessus, et le besoin de protéger les internautes, il devient important de fournir un système personnalisé, qui permet de partager les problèmes, d'interagir avec un système et de trouver des recommandations.

Pour cela, ce travail propose « Cyberhelper », un système de recommandation de cybersécurité personnalisé basé sur des questions pour la sensibilisation à la cybersécurité.

De plus, la plateforme proposée est équipée d'un algorithme hybride associé à trois différents algorithmes basés sur la connaissance, les utilisateurs et le contenu qui garantit une recommandation personnalisée optimale en fonction du modèle utilisateur et du contexte. Les résultats expérimentaux montrent que la précision obtenue en appliquant l'algorithme proposé est bien supérieure à la précision obtenue en utilisant d'autres mécanismes de système de recommandation traditionnels. Les résultats suggèrent également qu'en adoptant l'approche proposée, chaque utilisateur peut avoir une expérience utilisateur unique, ce qui peut l'aider à comprendre l'environnement de cybersécurité.

Mots clés : système de recommandation personnalisé, cybersécurité, utilisateur, cyberattaques

Abstract

With the proliferation of the virtual universe and the multitude of services provided by the World Wide Web, a major concern arises: Security and privacy have never been more in jeopardy. Nowadays, with the Covid 19 pandemic, the world faces a new reality that pushed the majority of the workforce to telecommute. This thereby creates new vulnerabilities for cyber attackers to exploit. It's important now more than ever, to educate and offer guidance towards good cybersecurity hygiene. In this context, a major effort has been dedicated by researchers, governments, and businesses alike to protect people online against hackers and cybercriminals.

With a focus on strengthening the weakest link in the cybersecurity chain which is the human being, educational and awareness-raising tools have been put to use. However, most researchers focus on the "one size fits all" solutions which do not focus on the intricacies of individuals. This work aims to overcome that by contributing a personalized question-based recommender system. Named "Cyberhelper", this work benefits from an existing mature body of research on recommender system algorithms along with recent research on non-user-specific question-based recommenders.

The reported proof of concept holds potential for future work in adapting Cyberhelper as an everyday assistant for different types of users and different contexts.

Keywords: Personalized recommender system, cybersecurity, user, cyberattacks

Contents

List of Tables

List of Figures

List of abbreviations

RS	Recommendation System
NLP	Natural Language Processing
KB	Knowledge Base
ML	Machine Learning
FPAN	Feedback-guided Preference Adaptation Network
GNN	Graph Neural Networks
LDA	Latent Dirichlet Allocation
CF	Collaborative Filtering
CBF	Content-Based Filtering
WHO	World Health Organization
VPN	Virtual Private Network
CDC	Centers for Disease Control and Prevention
WSD	Word sense disambiguation
NER	Named Entity Recognition
POS	Part of Speech

To my father, M. Both Pierre, gone too soon this year.

Thank you for your endless love, support, and encouragement

When I was down, you'd been there holding me up, spread your wings as you go,

and when Gad takes you back, he says Hallelujah, you were home…

Acknowledgments

I would like to express my sincere thanks and gratitude to my advisor, Professor Esma Aïmeur, for her patience, understanding, immense guidance, wonderful supervision, and full support. I was extremely grateful to have an advisor who cared so much about my work and who was always available to respond to my questions and queries promptly. Thanks for encouraging me and giving me some advice throughout my time as her student. Without her continuous guidance and incredible support, this dissertation would not have been possible.

I would like to thank, with a lot of gratitude, Professor Hicham Hage for his precious time, his guidance, his support, and immense knowledge.

Moreover, I would like to thank the jury members, Professor Claude Frasson and Professor Yoshua Bengio, for spending their precious time in order to review and evaluate my thesis.

I would like to express a special thanks to my fellow labmates Rim for her patience, time, for guiding me, and for giving me some advice while I write my research work. I would like to thank Ramya, Yishu for being fun deskmates when we were together. Thanks to Ayman for explaining cyber-security aspects and helping me on many occasions. Thanks to Dorsaf and Fahed for the time we spent together. I would like to thank Mme Céline Bégin for her advice and her efficiency as well as the whole UdeM administration team.

I would also like to express all my gratitude and thanks to my family, my friends Marie-Caroline, Sati, Maeva, my boyfriend Etienne, for all the unconditional support and encouragement that they gave me during this project.

Last but not the least, I would particularly like to thank my late father M. Both Pierre, gone too soon early this year, He was my biggest motivation during this research work.

Chapter 1 – Introduction

This chapter presents the context of our research work, explains the problem addressed in this thesis, identifies the research questions and objectives. Finally, point out the objective of our research work.

1.1 Problem Definition and Motivation

In the new era of technology and the Internet, information such as in identity cards, health records, or politicians' decisions, etc. is becoming available online and can cause real damage if used by unauthorized entities (Yusri, 2020). A survey conducted in early April 2020 by the Pew research center indicates that during the Covid19 outbreak, more than 53% of Americans say the internet has been essential for them (Emily Vogels, 2020). The alarming part is that over 45% of participants confessed to using the same passwords for different applications, and 90% admitted to using devices provided by their companies for private purposes (Proofpoint, 2020). Such behavior does not respect the cybersecurity hygiene recommended by the cyber experts.

Modern technologies in some cases increase the vulnerabilities of internet users. In fact, internet privacy becomes a big concern for many people of all ages, as companies and organizations track our behavior online over time. The author Sicari (2020) notes that the use of 5G technology generates big concerns in terms of security and privacy in our everyday lives. Moreover, Tesfay (2019), has noted a lack of transparency on the part of companies and service providers and the authors recommend that users should be informed about their data collection and processing practices through clear privacy policies. (Salema, 2020b) discusses the challenges of mitigating self-disclosure by proposing a platform to provide personalized recommendations taking into consideration the preferences of each individual. During this Covid 19 outbreak, people are working remotely and using unsafe devices that make them vulnerable to cyber attacks. As a result, the number of cyberattacks has peaked and that includes but is not restricted to hacking, malware-laced email phishing, scammers posing as corporate help desks, and malware in COVID-19 information sites (Daniel Mikkelsen, 2020).

According to Canadian Anti-Fraud (Canada.ca, 2021), the number of victims of cyber attacks has significantly increased since the beginning of the pandemic. Between March 6, 2020, and March 31, 2021, more than 15,198 Canadians were victims of COVID-19 fraud. This number is alarming and reinforces the urgent need for cybersecurity assistance to help people understand and avoid online dangers and be as protected as possible while using online services. The preservation of users' privacy is an important criterion that every company or business must consider before implementing any application or software but an unaware or careless user can cause damage despite that.

With a focus on the human side and how to guide users in a customized way, personalized recommender systems present the main topic of interest.

Within this scope, there are many elaborations on the techniques (Balog, 2019) and algorithms that generally fit in one of the following categories: collaborative filtering, content-based filtering, or hybrid systems.

What makes the system personalized is the integration of users preferences and interests, intent prediction (*Pan et al., 2020*), topic modeling (*Jelodar et al., 2019*), sentiment analysis, and opinion mining (*Sundermann* et al., 2020) are used as key factors to find similarity between users (*D. Wang et al., 2020*). However, each research adopts only a small set of factors of user preference and interest, which are not sufficient for a system such as our Cyberhelper that aims to assist with the user and context-specific security problems. The aforementioned issue is what motivated this work that aims to address the following points:

- How to detail a user model and what can be considered as the context, both of which are inputs for the system?
- How to adapt generic existing recommender systems algorithms into a Cybersecurity personalized and context-aware system?
- How to back up the potential of our Cyberhelper through a proof of concept?

1.2 Research Objectives

The purpose of this thesis is to provide a personalized solution to users who are facing a cyberattack. The system helps the users to understand, prevent, and solve some cyberattacks that they experience in this Covid 19 pandemic time. More specifically, the objectives of this thesis are:

- Propose a well-defined user model and context that cover the necessary concepts of user modeling and context modeling such as personal information, location, time…etc.
- Design a cybersecurity knowledge-based database that fits with the actual cybersecurity landscape in this Covid19 pandemic.
- Adapt the existing recommendation algorithms and the design of generic question-based recommender systems into our personalized context-aware Cyberhelper.
- Provide a proof of concept of our Cyberhelper that paves the way for potential future work on personalized cybersecurity assistants.

1.3 Main Contributions and their Originality

Motivated by the importance of keeping good cybersecurity hygiene in this Covid19 pandemic time, and the importance of privacy awareness online, a personalized question-based cybersecurity recommendation system called Cyberhelper is proposed to support users who experience cyberattacks. Knowing the different backgrounds that users might belong to, one of the personalization criteria is the fact that the system is designed for people with high, medium, or no cybersecurity knowledge.

This work consists of the following contributions:

- Propose a personalized question-based cybersecurity system based on existing generic question-based recommenders and simple yet efficient algorithms.
- Report a proof of concept of the Cyberhelper with encouraging potential for future development.

1.4 Thesis Structure

- Chapter 1 presents the main research directions, such as research background, problem statement, and research objectives and questions. Also, this chapter explains the research scopes, research significance, and the definitions of key terms.
- Chapter 2 reviews the theoretical and empirical considerations of this research. The directions of Personalized question-based cybersecurity recommendation systems would be discussed in this chapter and the techniques applied in this domain.
- Chapter 3 introduces the data preprocessing techniques used in this research. Moreover, the steps of the proposed Personalized question-based cybersecurity recommendation system.
- Chapter 4 presents the experimental evaluation techniques used to evaluate the proposed system.
- Chapter 5 summarizes the research results based on the research objective and clarifies the research contributions and implications. It also suggests future works.

Chapter 2 – Background and Literature Review

This chapter provides an overview of the theoretical and empirical considerations in personalized recommendation systems. After presenting the previously published research in this field, this chapter moves on to more details about the methodology behind our Cyberhelper.

2.1. Personalized Recommendation Systems

This section presents the studies that have been done in the field of personalized recommendation systems (Betancourt, 2020). The development of the new internet network era provides a huge amount of data and contributes to a lot of information overload. In the real-world scenario, users used to browse irrelevant information before getting the correct information about their request. Therefore, there is a huge demand for platforms to provide services based on personal users' interests and preferences. In this case, a personalized system would ideally be a part of the solution that reduces the "information overload". A personalized recommendation system is not a new topic, and many online platforms have been designed to improve the user experience. For example, authors (Hui, 2019) propose a "personalized intelligent educational system" designed to serve students, teachers, and other staff. It consists of tailored recommendations of teaching resources that help them to get the right multimedia tools they need. Based on this work, the system increases the number of learners and tracks their characteristics through feedback. Table 1 contains various examples of recommendation systems technologies used by different platforms.

Table 1: Example of Recommendation systems

Name	**Domain**	**Type**	**Year**	**Technique**	**Reference**
Amazon	Ecommerce	Item-to-Item Collaborative Filtering	1994	It is based on three factors: i) Ratings, ii) Buying behavior iii) Browsing behavior of the user. uses neural networks.	(*Linden et al., 2003*)
MovieLens	Movie	Collaborative filtering	1997	Based on movie ratings, it generates personalized predictions for movies.	(movielens.org)
Netflix	Movie	Hybrid recommendation system	1997	Netflix is based on Machine learning, reinforcement learning, matrix factorization, neutral work.	(Gomez-Uribe & Hunt, 2015)
Google	Advertisement, News	Hybrid recommendation system	1998	It uses an ML-based recommendation model and matrix factorization.	*(Serrano-Guerrero et al., 2011*)
TripAdvisor	Travel	Content-based filtering	2000	It depends on defining an appropriate similarity measure. It uses Pandas in Python and k-means visualizations	(*Nilashi et al., 2018*)
Pandora	Music	Content-based approach	2000	It is based on a multi-tiered approach and uses personalized filtering based on the user's choice.	(Howe, 2009)
Last. fm	Music	Collaborative filtering	2002	It depends upon the relations between users with similar preferences. It uses the technique is crowdsourcing scrobbling.	(Schedl, 2016)
LinkedIn	Job	classical collaborative-based filtering recommendation	2003	Linkedin has used Amazon's Mechanical Turk for building the crowdsourcing and skill inference algorithm.	(*Domeniconi et al., 2016*)
Facebook	Social Network	Collaborative filtering	2004	Facebook uses a ball tree data structure, based on Apache Giraph. It also uses Matrix factorization.	*(Forouzandeh et al., 2018*)
YouTube	Entertainment	Collaborative filtering	2005	It uses the batch-oriented precomputed approach and deep neural network	(*Covington et al., 2016*)
Spotify	Music	Collaborative filtering	2006	It depends on three models: i)Collaborative Filtering, ii)Natural Language Processing iii)Audiomodels	(*Millecamp et al., 2018*)
Twitter	Social Network	Hybrid RS	2006	It is based on the hashtag ranking method.	*(Hannon et al., 2010*)
Airbnb	Travel	Content-Based RS	2008	It is based on a search algorithm that supports deep learning and neural networks.	(Wu & Grbovic, 2020)
Instagram	Social Network	Rule-based personal assistant	2010	It's based on machine learning techniques. There are three main factors of Instagram i.e. interest, recency, and relationship.	(Huang & Wang, 2017)
Kindred Works	Ebook	Content-based RS	2012	It has mainly two concepts: user interface and application programming interface	(oclc.org)
Tinder	Dating	Collaborative-based filtering	2012	It is based on TinVec and Word2Vec mechanics. Tinder uses GPS technology.	(Andrews, 2015)

Uber Eats	Restaurant	Friends' recommendation	2014	It is based on the representation learning approach, GraphSAGE, candidate generation, and personalized ranking.	(*Jain et al., 2019*)

According to the presented examples, it's important to mention that multiple recommender engines use Collaborative filtering and Hybrid recommendation system techniques rather than content-based filtering recommendation techniques.

User interest and preference are considered essential aspects of personalized recommendation systems. It includes various factors about the user modeling and how to meet the user's particular demand. In recent years, many researchers focus on user-interest modeling and achieve promising research results. For example, He (2014) presents how to use Latent Dirichlet Allocation (LDA) to determine user interest. The main idea behind the user's interest and preference model is to understand the goal they want to achieve and predict the user's intent to provide the help they need.

One of the most recent articles in this field (Xu, 2021) focuses on user preference estimation and proposes a Feedback-guided Preference Adaptation Network (FPAN) that applies a Graph Neural Networks (GNN) to embed the users, items, and attributes to adapt to the user preference and feedback. However, personalized systems need to collect the user's private information. Privacy is a big concern in the modeling of personalized recommendation systems. Therefore, it is essential to consider the integration of privacy concerns into the personalized recommendation systems. In the article "privacy-preserving personalized recommendation system", the authors (Ammad-Ud-Din, 2019) introduce an innovative method to protect user privacy without collecting and storing the user personal data using a Federated Collaborative Filtering method. This method is based on the stochastic gradient approach, consisting of carrying out the model update and distributing parts of the model computation to the client.

As an empirical study, (Salema, 2020a) designed a personalized harm-aware recommender system. The platform analyzes the user's risk of disclosure and then recommends the user to reduce that risk, and if there is a disclosed data, it generates the threat vector. This study showed that users could be guided to preserve their privacy by letting them know when there is a risk of disclosing their personal information through nudges as a tailored recommendation.

The e-learning personalized recommendation systems (Benhamdi, 2018) provides a framework where students can get the best learning materials according to their preferences, interests, background knowledge, and their memory capacity to store information. Klavsnja-Milicevic (2018) applies collaborative tagging techniques based on user preference, with the main goal to improve the learning process and better classify the learning content. Further mention and discussion of recommender systems techniques and algorithms are presented in the next section.

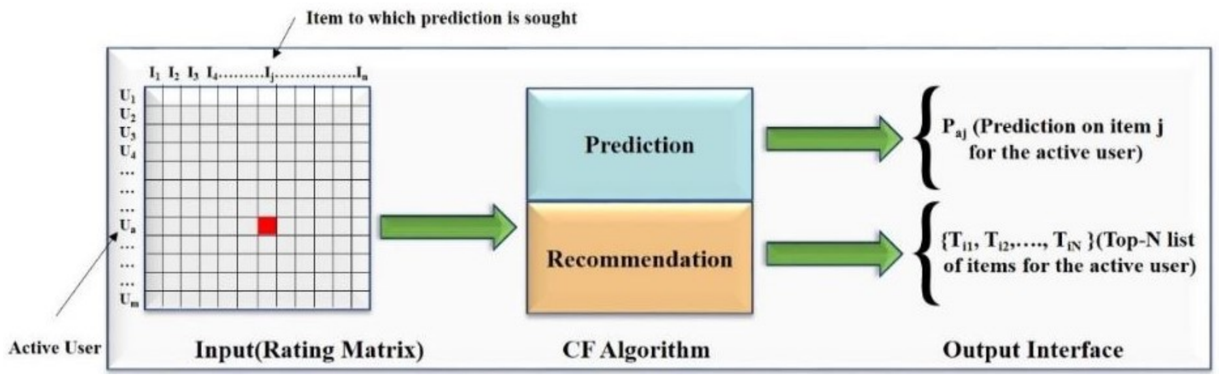
2.2. Personalized Recommendation Systems Technologies

Traditional recommendation systems are based on three main approaches: Collaborative filtering, content-based filtering, and hybrid approaches as seen in Table 1. However, it’s important to mention the difference between traditional and personalized recommendation systems. In our case, it’s important to stay focused on techniques used in the personalized recommendations but also ones that have a solid mature foundation.

The following section discusses the main characteristics of collaborative filtering, content-based, knowledge-based and hybrid recommenders. In addition to that, a comparative table of the different recommenders' techniques will be included.

2.2.1 Collaborative Filtering technique

Early research in the recommendation system field was performed by Goldberg and his colleagues (1992): they introduce the Tapestry, an experimental mail system that was built to support collaborative filtering techniques. In this recent years, various recommendation systems technologies have been introduced, and among those systems, the most popular is collaborative filtering (*Geuens et al., 2018*). This system is based on the association of users or customers with the same interests or opinions. Collaborative Filtering calculates the matrix of similarity between users and computes it with the user’s historical information to get the recommendation. . Figure 1 shows an example of a collaborative filtering process.



Item to which prediction is sought
I_1 I_2 I_3 I_4.........I_j...............I_n
U_1
U_2
U_3
U_4
U_a
U_m
Active User
Input(Rating Matrix)
Prediction
Recommendation
CF Algorithm
P_{aj} (Prediction on item j for the active user)
$\{T_{i1}, T_{i2}, \ldots, T_{iN}\}$(Top-N list of items for the active user)
Output Interface

Figure 1: Collaborative filtering process (Bokde et al., 2015)

The recommendation is based on the users with similar interests or preferences *(Schafer et al., 1999*). Such systems are inspired by Tapestry (*Goldberg et al., 1992*) or Grouplens *(Konstan et al., 1997*), they are based on ratings. The ratings are used to match users with the same preferences.

The collaborative filtering algorithms are either memory-based or model-based algorithm (Alhijawi & Kilani, 2020). In the memory-based algorithm, the system calculates the ratings using the average rating of similar users and then recommends items with the highest rating. The process is as follows: they calculate the Nearest Neighboring user, which is usually found through the Pearson Correlation Coefficient *(Edelmann et al., 2021*) or the cosine similarity. In the model-based algorithm, we consider all the ratings as an input that will be used to building the model for predicting ratings using a Bayesian Network. As shown in Figure 2, based on the user's movie-watching history, user *a* and user *b* share preferences, so the userCF can recommend movie 3 and movie 6, to user *b* according to what the user *a* likes. in the meantime, itemCF recommends movie 2, to user *b*, which is similar to movie 6 that user *b* liked before.

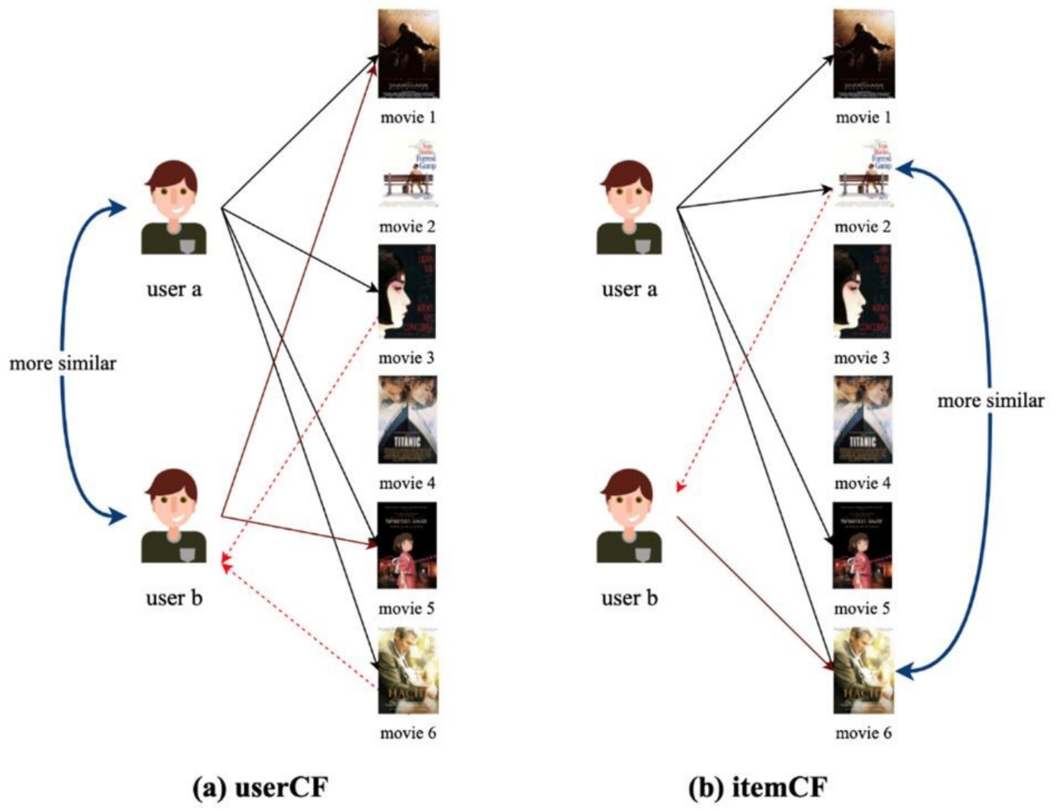


Figure 2: Demonstration for collaborative filtering (CF) algorithm (*Yang et al., 2021*).

Collaborative filtering has some limitations: The well-known (1) "cold start" problem (Rabaa Alabdulrahman & Paquet, 2019), it is hard to recommend accurately a new user without any preference. Alternately, the new item is also difficult to recommend because it requires ratings by the users. (2) The data sparsity problem happens when people rate similar items and many items get very few ratings, which leads to a sparsity of data. (3) The gray sheep problem occurs when users are interested in unique or rare items which are hard to categorize.

2.2.2 Content-Based Filtering technique

The content-based technique is based on the items with similar contents that the users like. It refers to the item-to-item correlation methods (*Schafer et al., 1999*). Assuming that if, in the past, user U_i liked item I_j, then in the future, he/she will like items similar to item I_n (Lu & Du, 2016).

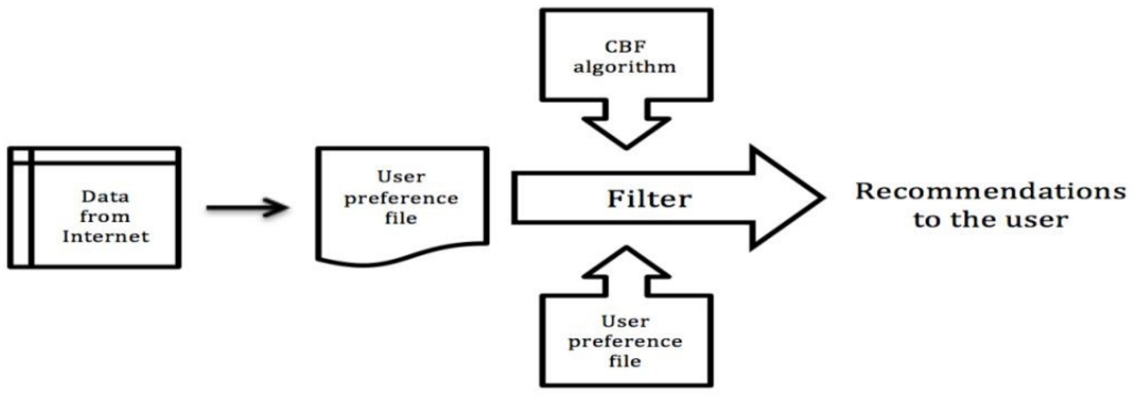


Figure 3: Content-based Filtering (*Jiang et al., 2010*)

The quality of the recommendation depends on the information available about the users and items in the system. Compared to collaborative filtering, content-based filtering does not have the new-item problem. In fact in content-based filtering with no learning (*Wang et al., 2018*), there is no need for information from other users in the recommendation process. This content-based filtering is more used in a system with a low number of users (*Kim et al., 2011*). However, machine learning-based content-based recommendations can use all the data.

Content-based filtering is facing two big challenges: limited content analysis and overspecialization. The limited content analysis refers to the lack of relevant information about items and users. On another side, overspecialization occurs when the system repeatedly recommends items from the same category. To solve this problem, it’s possible to add various items to the recommended lists (Madadipouya & Chelliah, 2017). Figure 4 shows an example of content-based filtering where users rate items and receive recommendations of items which is similar to those who receive a good evaluation from the current user (Ua).

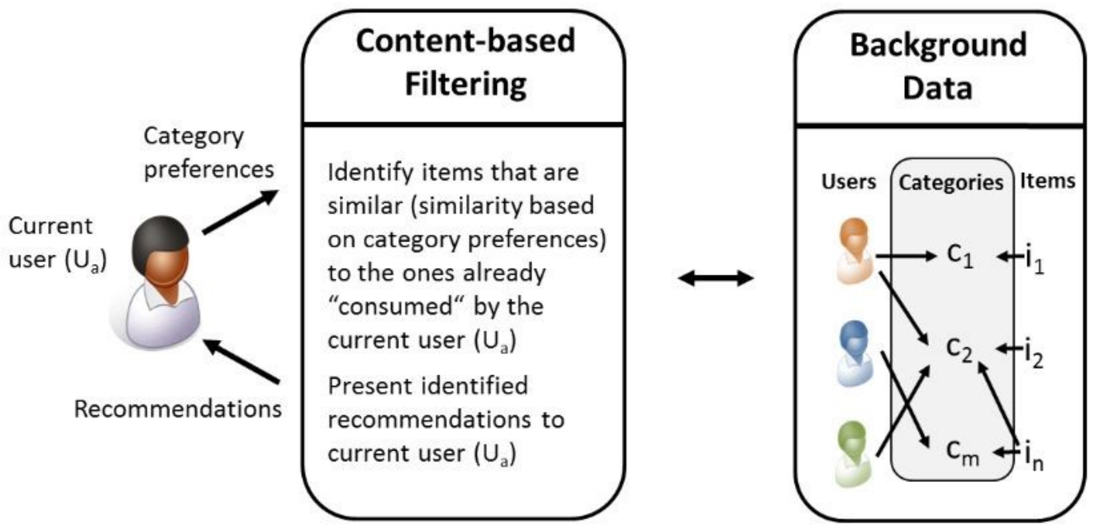

Content-based Filtering
Identify items that are similar (similarity based on category preferences) to the ones already "consumed" by the current user (Ua)
Present identified recommendations to current user (Ua)
Background Data
Users
Categories
Items
Current user (Ua)
Category preferences
Recommendations
c1
c2
cm
i1
i2
in

Figure 4: Demonstration for content-based filtering *(Felfernig et al., 2014)*

Despite the challenges mentioned above, Content-Based Filtering techniques are used to fix some of the Collaborative Filtering drawbacks such as sparsity, cold-start, and scalability *(Kim et al., 2011)*. Table 2 summarizes the advantages and disadvantages of each type of personalized RS approach mentioned above.

Table 2: Recommendation systems, advantages, and disadvantages (Alabdulrahman, 2020)

Technique	Advantage	Disadvantage
Collaborative Filtering	• Easy to analyze content • Able to perform with limited content • can improve quality over time	• Scalability • Computational cost with a large database • Cold-start problem • Grey sheep problem • Data Sparsity problem
Memory-based CF	• Low-cost and easy implementation • Shorter computational time. New data can be added incrementally • Considers the content of items to be recommended	• Requires scanning the entire data • Computation complexity • Limited scalability for large and complex datasets • Data sparsity causes low performance • Dependent on human ratings • Cannot handle cold-start users and items
Model-based CF	• Can handle data sparsity • Improves CF performance • Treats scalability • Shortens computational time	• Offline in nature
Content-based Filtering	• No need for other users' profiles • Fast adjusting user profile change • Overcomes cold-start problem • Quality can improve over time	• Limited content analysis. Overspecialization • Two items with similar features

Personalized recommendation systems are an emergent field over the last decade, a lot of collaborative filtering and content-based filtering techniques are used by many companies, researchers to increase their performance and reach more and more users. Table 2 has shown their strength as well as limitations, and to fix these issues, researchers introduce the hybrid filtering techniques.

2.2.3 Hybrid Filtering Technique

The Hybrid Filtering algorithms combine collaborative filtering and content-based filtering and other recommendation techniques (*Lu et al., 2015*). To combine CF and CBF, some researchers propose to implement each technique independently. Others use CF characteristics into CBF, vice versa, another integrates both techniques into the same model and then combines their predictions.

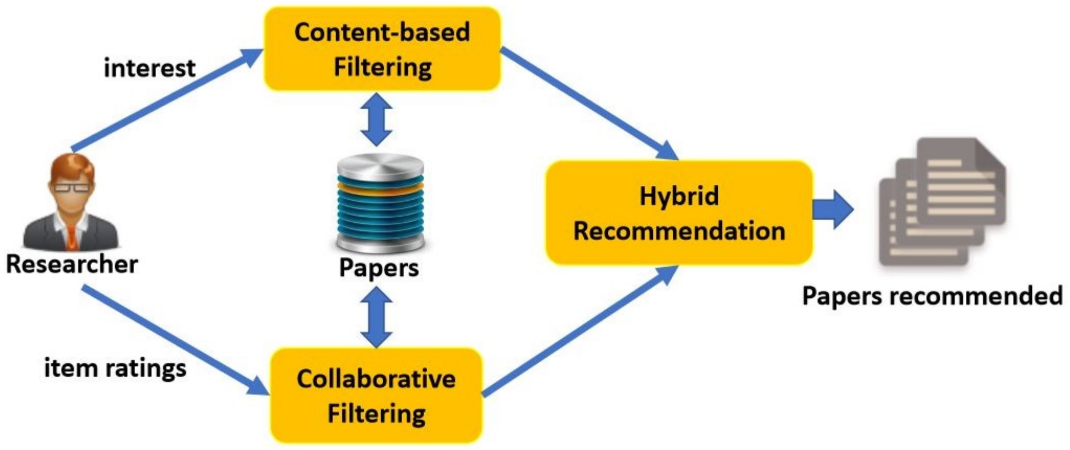


Figure 5: Example of Hybrid filtering system (*Bai et al., 2019)*

The author Robin Burke (2002) presents seven hybridization techniques: (a) A weighted hybrid technique combines the results of two independent operating recommenders in one score. (b) A switching hybrid recommender selects the best recommendation for a particular situation. (c) A mixed hybrid technique presents the results of two or more recommender techniques within one standard result list. (d) A feature combination uses features from different knowledge sources as an input to a single recommendation technique. (e) A feature augmentation uses one recommendation technique as input to another one. (f) A cascade uses the output of one recommendation technique as input to the next one. (g) meta-level uses the resulted model of one technique as an input to another one (Kumar & Thakur, 2018). All these techniques aim to provide better accuracy of the recommendation. Consequently, the use of these hybridization techniques depends on the domain and application of the Recommendation Systems.

Table 3: Example of Personalized recommendation Systems

Topic	Items	Data used	Recommendation approach	Supporting method	Data mining technique	Reference
An Interactive Personalized RS Using the Hybrid Algorithm Model	Product	Customer's feedback information	Hybrid algorithm of CBF, item-based CF, user-based CF, and association rules	Consumer coverage	Data mining techniques	**(Guo, 2017)**
DuerQuiz: A Personalized Question RS for Intelligent Job Interview	job	Recruitment dataset (job posting, resume, experience sentence)	Skill-graph	LDA, Gradient Boosting Decision Tree (GBDT)	LSTM-CR, part-of-speech (POS), word embedding	**(Qin, 2019)**
Deep Learning Recommendation Model for Personalization and RS	product	Random, synthetic, and public data sets	Deep learning recommendation model (DLRM)	Matrix Factorization, multilayer perceptron, linear and logistic regression	Latent factors, embedding	**(Naumov, 2019)**
An LSTM based model for personalized context-aware citation recommendation	citation	Scientific papers	LSTM	Convolutional neural networks (CNN)	word embeddings, CBOW	**(Yang, 2018)**
Personalized RS Based on Collaborative Filtering for IoT Scenarios	product	User rating, tag, number of users, and products (book, movie)	Collaborative filtering	K-means, time correlation coefficient	Clustering	**(Cui, 2020)**
An intelligent learning system based on personalized recommendation technology	learner	Learning resource	hybrid recommendation algorithm	SVM, Pearson correlation coefficient,	Any data mining technique	**(Li, 2019)**

To summarize, Table 3 presents multiple personalized recommendation systems, their domains, features, and techniques behind their systems. It is important to note that every personalized recommendation system changes its algorithms based on some factors such as the user needs/interests, or the domain of the applications. And the results are still quite good. Also, there is a rapid development of new techniques in the algorithms to solve the problems faced by the users and improve the performance of the systems. Many companies invest a huge amount of money to develop their platforms where every user will have a unique and personalized experience that can lead to an increase in the quality of the recommendations.

2.3. Cybersecurity attacks

Early research in the cyberattack domain was performed by Robert Morris (Eisenberg, 1989), who launched one of the first computer worms to map the Internet. Over the years, Cybersecurity attacks significantly have increased in terms of frequency and sophistication. This significant peak in the number of attacks encourages more development and innovation in detection and prevention strategies. Traditional methods of detection and prevention are no longer sufficient to solve the new forms of cybersecurity attacks.

Some researchers are studying how technological changes can influence the identification of new varieties of threats (Kaloudi, 2020). Brundage et al. (2018) present different scenarios to show the various attacks that people will see in the next years. They predict that the rapid adoption of AI technology would significantly change the current cybersecurity attack landscapes and improve the existing attacks with a more precise target. There are new forms of attacks with different attributes, characters, and the cyber attackers constantly improve their cyberattacks strategies to exploit users' vulnerabilities, and misconfiguration of IT systems.

With the outbreak of Covid 19, cyber attackers are more inspired to take advantage of people's vulnerabilities when they are not well prepared or without cybersecurity knowledge. At the beginning of the pandemic, from February to March 2020, there were a more than 220 times increase in spam emails and 260% in malicious URLs. In the meantime, until April 2020, more than 907K Spam messages, 737k Malware attacks, and 48k hits on malicious links were reported around the world (TrendMicro, 2020).

Phishing remains the most used method of attackers to conduct cyberattacks. PhishLabs (PhishLabs, 2018) report that in 2017, over 70% of security breach attacks associated with nation-state or state-affiliated actors involved phishing. To effectively detect and combat cyber attacks (Ferrag, 2020), it is essential to truly understand cyber-attack operations.

2.3.1 Phishing Attacks

The Covid19 pandemic made phishing attacks easier than usual. The cyber attackers take advantage of this situation and the vulnerability of users and organizations to target their systems and devices. There have been many cases where phishing attacks are spreading on Facebook Twitter, Whatapps, and many other applications. Also, early in February 2020, there have been observed various coronavirus-related emails with malicious attachments were sent to users. Moreover, in many cases, the attackers pretended to be from legit services and organizations such as international organizations, hospitals, Banks, …etc. For example, the World Health Organization (WHO) was hit by a phishing attack (WHO, 2021). The attacker has used domain spoofing (coronavirusfund@who.org) to fool the victim and pretend that the email is coming from WHO, so they can easily ask them to donate in bitcoins, etc. Over the years, phishing attacks (Benavides, 2020) have become one of the most common attacks faced by internet users. Typically, in phishing attacks, the attackers tend to deceive their victims to give out sensitive information such as user's passwords, bank statements, health records, or other personal information. The most common phishing-based attack techniques use by attackers to mislead people are spoofed emails (Kumar, 2020) or fake websites. The preparation of more realistic and sophisticated phishing kits on the cyber underground may be encouraging a renewed interest in this form of attack. In 2020, (Symantec, 2020) reports an increase of approximately 80% of phishing attacks since the beginning of the Covid 19.

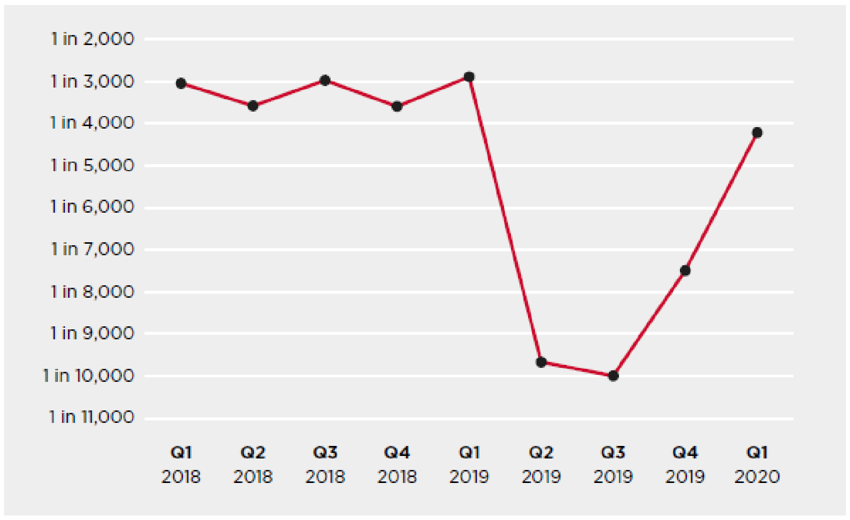
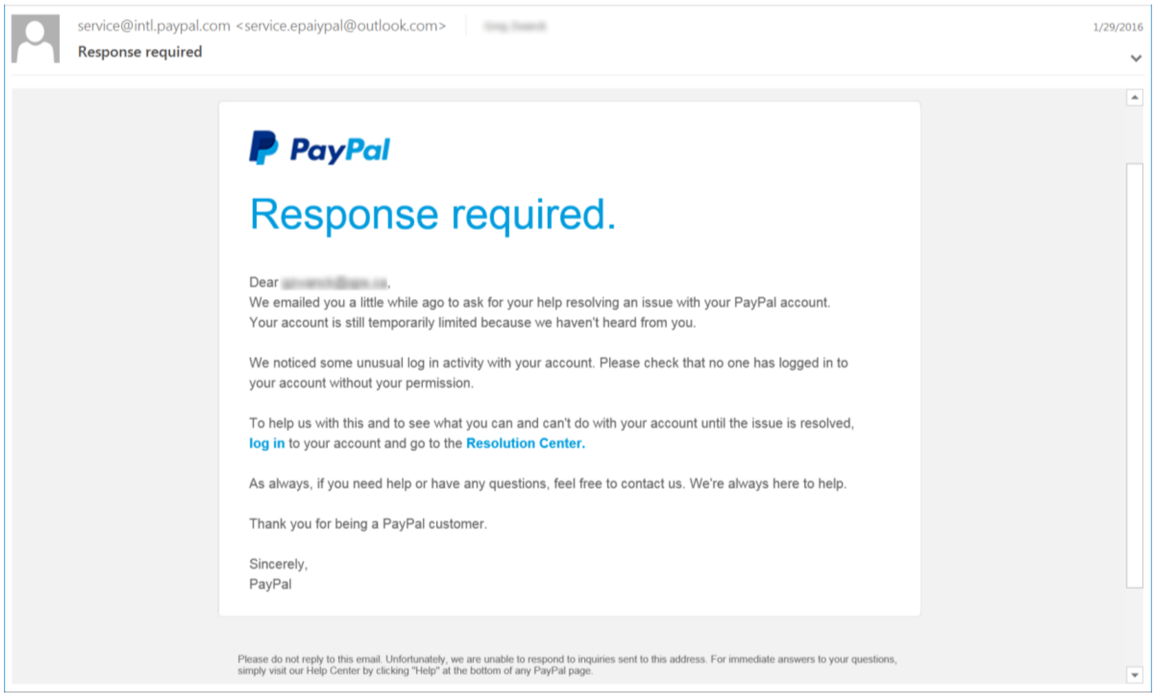


Figure 6: Phishing rate (Symantec, 2020)

Usually, it is easy for the attackers to reach their victims using a phishing attack (Mao, 2018) because users don't have a real idea about which website they can trust. The use of phishing websites is frequent in phishing attacks (phishing.org, 2021), in this case, the attackers create fake website pages by copying legitimate websites and send suspicious URLs to the targeted victims through spam messages, emails, texts, or online social networks.



service@intl.paypal.com <service.epaiypal@outlook.com> [redacted] 1/29/2016

Response required

PayPal

Response required.

Dear [redacted],
We emailed you a little while ago to ask for your help resolving an issue with your PayPal account.
Your account is still temporarily limited because we haven't heard from you.

We noticed some unusual log in activity with your account. Please check that no one has logged in to your account without your permission.

To help us with this and to see what you can and can't do with your account until the issue is resolved, log in to your account and go to the Resolution Center.

As always, if you need help or have any questions, feel free to contact us. We're always here to help.

Thank you for being a PayPal customer.

Sincerely,
PayPal

Please do not reply to this email. Unfortunately, we are unable to respond to inquiries sent to this address. For immediate answers to your questions, simply visit our Help Center by clicking "Help" at the bottom of any PayPal page.

Figure 7: A phishing email that seems to come from Paypal (Phishing.org, 2020).

This is a phishing email, as appeared in a message from Paypal. The recipient is then advised to login with their credentials information to solve his/her Paypal issue.

Table 4: Type of phishing attacks with symptoms and impacts

Phishing attack					
Type	**Characteristics**	**Scope**	**Challenge**	**Symptoms**	**Impacts**
Standard Phishing	Pushes targets to disclose sensitive and valuable data.	Targeting a large number of users.	Encouraging users to take steps and protect themselves. Phishing techniques are constantly being innovated, which makes them difficult to detect.	• Fake domain names • Grammatical errors in emails. • Email containing suspicious links or attachments. • Email asking for credentials or payment	• Financial damage • Reputational damage: the company may lose customers • Compromise of confidential data
Spear Phishing		Focusing attacks on organizations or specific individuals			
Whaling		Focusing attacks on C-level management			

It is seen that phishing emails remain the number one for phishing attacks. Attackers use a new variety of attacks to make them more credible and easier to get personal information from their victims. With the Covid19 pandemic, there is an increase in SMS texting attacks (smishing) or the use of messaging on social media and gaming platforms.

2.3.2 Malware

The current situation gives many advantages to cyber attackers in targeting vulnerable people and organizations by spreading Malware (*Han et al., 2017*), Trojan, Spyware, ...etc. through websites and emails. The main tactic used to deceive users is by clicking on the link or downloading the malware from spam emails. In fact, 94% of computers corrupt by malware were infected by an email (WHO, 2021). During this pandemic time, malware emerged as a major threat to cybersecurity, attackers can easily infiltrate and infect the systems directly. Some intelligent devices and smartphones are more vulnerable to malware attacks (TrendMicro, 2020) such as worms, backdoors, viruses, and trojans. An infected device can generate multiple malicious activities such as social engineering attacks which can cause the lost sensitive information, unwanted billing, make the device unavailable, …, etc.

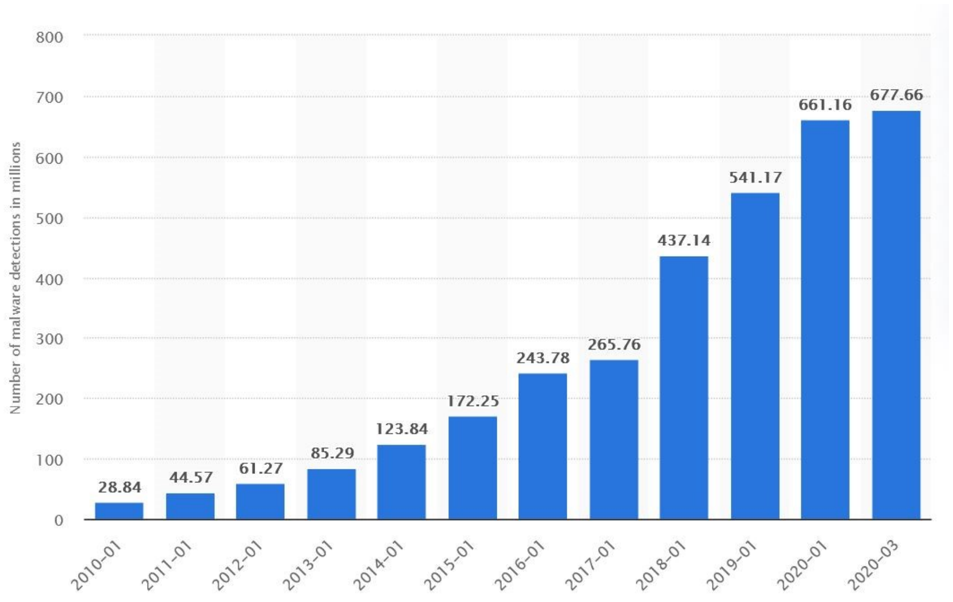
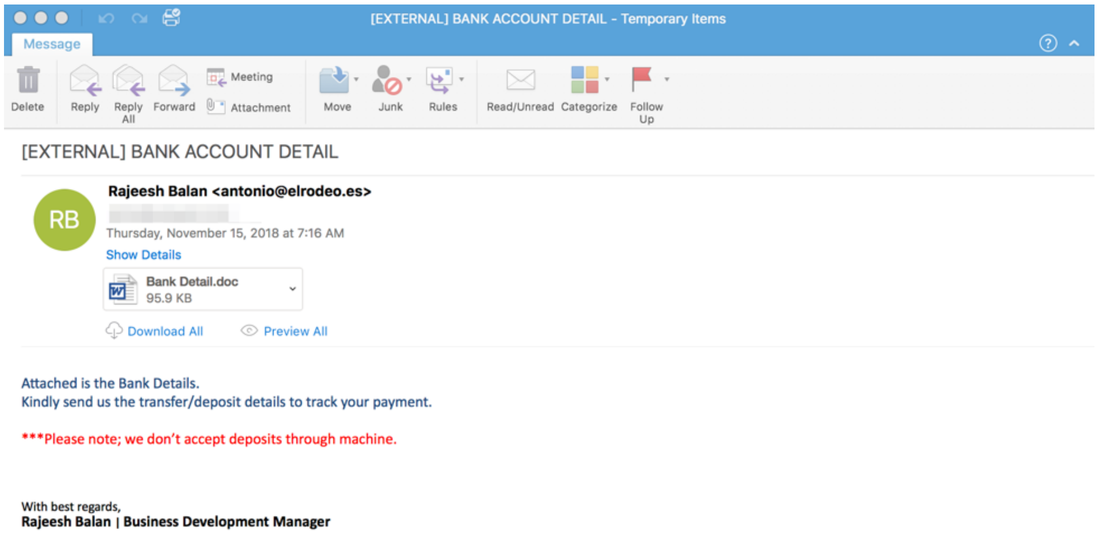


Figure 8: Development of malware worldwide 2015-2020 (Johnson, 2021)

Statista (Johnson, 2021) shows the higher growth of malware detection worldwide in the graph in Figure 8. Understanding different malware behaviors could be beneficial to the development of prevention techniques (Raghuraman, 2020) on computer systems.



[EXTERNAL] BANK ACCOUNT DETAIL

Rajeesh Balan <antonio@elrodeo.es>

[redacted]

Thursday, November 15, 2018 at 7:16 AM

Show Details

Bank Detail.doc
95.9 KB

Download All   Preview All

Attached is the Bank Details.
Kindly send us the transfer/deposit details to track your payment.

***Please note; we don't accept deposits through machine.

With best regards,
Rajeesh Balan | Business Development Manager

Figure 9: Example of a Malware attack by email (Schwarz, 2019)

In this example, the email contains bank detail ".doc" attachments. The document contains macros that, when enabled, download and execute the ServHelper malware.

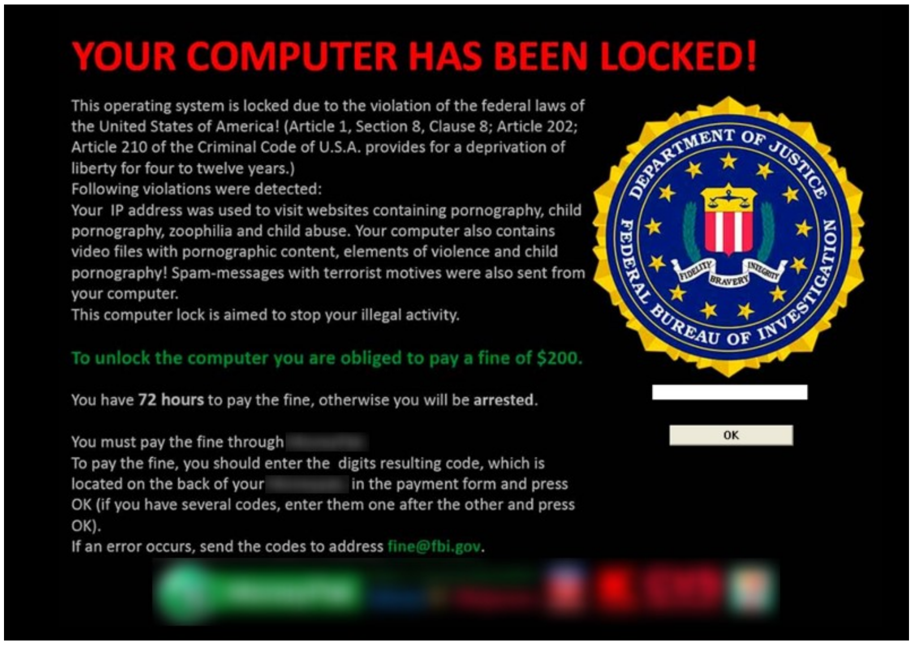
Table 5: Type of malware attacks with symptoms and impacts

Malware attack					
Type	**Characteristics**	**Scope**	**Challenge**	**Symptoms**	**Impacts**
Virus	Malicious program able to self-replicate into other executable programs to either exploit vulnerabilities in the compromised system or cause damage to it.	Requires a host program that must be actively executed by a user.	Detect and mitigate new viruses	•The system continuously crashes • Slows computer performance and unusual experience when surfing on the Internet. • Difficulties with the hard drive. • Computer storage is inaccessible or corrupted. • Unusual error messages. • Encrypted files or programs.	• Data breaches: Unauthorized access to confidential data • Downtime: Attacks can break down the company network infrastructure which will have severe effects on its operations • General disruption of services by taking control of system-critical programs • Money loss
Worm	Uses self-replication into other systems; targets the entire networks to create large botnets	The stand-alone program which does not need host files or user interaction	Spread quickly and very easily over a network.		
Ransom ware	Blocks users' access to their system or personal files and requests payment to regain access.	Requires other cyberattacks to spread.	Ransoms are hard to track, as almost all payment requests are made in cryptocurrency.		

Based on what happens in 2020, the malware attacks are expected to continue increasing and targeting individual or large enterprises. Attackers and cybercriminals change their tactics to focus on discrete infections through IoT and emails.

2.3.3 Ransomware

In this Covid19 pandemic time, many hospitals, medical centers, and public institutions face ransomware attacks (Interpol, 2020). In fact, cybercriminals are more confident using ransomware attacks against organizations or businesses because they know that those organizations cannot shut down their systems. They have to negotiate how to pay the ransom. Ransomware (Pope, 2016) is a type of malware that allows cyber attackers to encrypt a user's files *(Richardson et al., 2021*) and then demand payment for the decryption key.


YOUR COMPUTER HAS BEEN LOCKED!

This operating system is locked due to the violation of the federal laws of the United States of America! (Article 1, Section 8, Clause 8; Article 202; Article 210 of the Criminal Code of U.S.A. provides for a deprivation of liberty for four to twelve years.)
Following violations were detected:
Your IP address was used to visit websites containing pornography, child pornography, zoophilia and child abuse. Your computer also contains video files with pornographic content, elements of violence and child pornography! Spam-messages with terrorist motives were also sent from your computer.
This computer lock is aimed to stop your illegal activity.

To unlock the computer you are obliged to pay a fine of $200.

You have **72 hours** to pay the fine, otherwise you will be **arrested**.

You must pay the fine through [illegible]
To pay the fine, you should enter the digits resulting code, which is located on the back of your [illegible] in the payment form and press OK (if you have several codes, enter them one after the other and press OK).
If an error occurs, send the codes to address fine@fbi.gov.

Figure 10: Example of a Ransomware attack (Zetter, 2017)

Some ransomware attacks displayed a pop-up message on the victim's machine saying that they are involved in Child porn activities or another kind of crime and demanded payment to remove it. The victim is instructed to make payments in 72 hours either through an SMS text message or by email that would earn the attacker revenue.

Most of the time, ransomware takes place with phishing campaigns or the use of spam emails that contain ransomware to bypass a user's technical safeguards and initiate restriction access. There are two main categories of ransomware (Brewer, 2016). The first one is the locker ransomware (Su, 2018), where the victim machine is locked. The second category is encryption ransomware which encrypts the victim's files. This pandemic favors the development of a new type of ransomware named "Coronavirus" (TrendMicro, 2020), which was spread through a phishing website *Wise learner*. The victims visited the website, downloaded the malicious setup file "WSHSetup.exe." and installed this file on their computer, allowing the malware to exfiltrate account information from web browsers, instant messengers, email, VPN, cryptocurrency, and steal the system information as well.

Another example of ransomware attacks at this Covid19 time is Covid Lock (Ransomware), a malicious Android app. This app locks victims' phones, threaten to delete phone data, leaks account information in social networks and give the victims 48 hours to pay ransom in bitcoin.

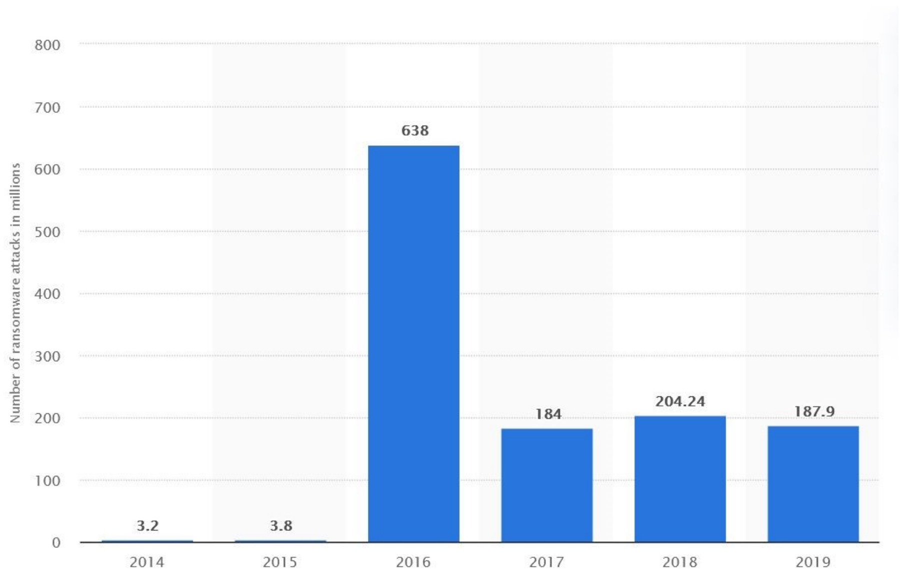


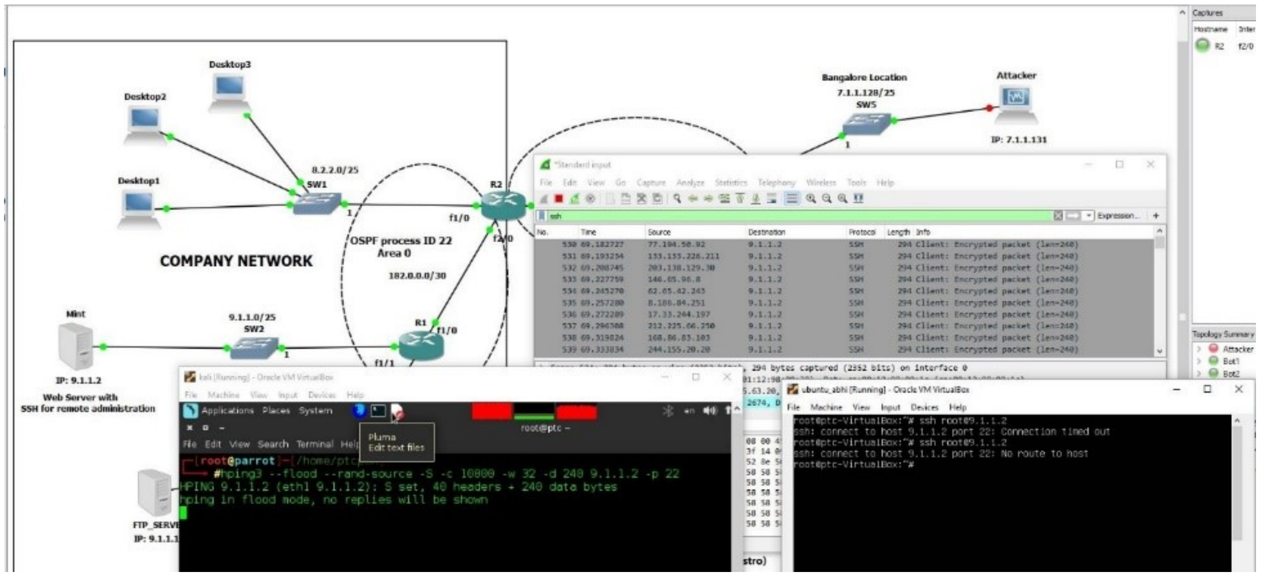
Figure 11: Number of ransomware attacks per the year 2014-2019 (Johnson, 2021)

Ransomware attacks have become more sophisticated and highly targeted against specific businesses, as well as local government organizations. Attackers are better prepared as they spend more time adjusting their ransomware tactics.

2.3.4 Denial of service attacks

With the Covid19 pandemic, most organizations and individuals have faced a considerable amount of Distributed Denial of service (DDoS) attacks *(Khan et al., 2020*). In today's internet era, Denial of Service (DoS) attacks are one of the most severe and complex attacks among the security problems. The cyber attackers target organizations' systems or websites and then flood them with multiple botnets to crash the regular traffic of those websites. This kind of attack can easily damage the communication resources of its victim and limit access to a machine or service instead of subverting the service itself. One of the most recent examples is a DDoS attack on the U.S health agency (Stein & Jacobs, 2020). The attackers targeted the U.S

Department of Health and Human Services' website by flooding millions of users at the same time.

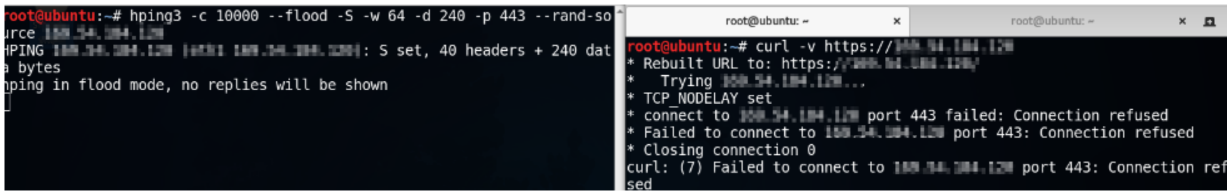


```
No.  Time       Source           Destination  Protocol  Length Info
530 69.182727   77.104.50.92     9.1.1.2      SSH       294 Client: Encrypted packet (len=240)
531 69.193254   133.133.226.211  9.1.1.2      SSH       294 Client: Encrypted packet (len=240)
532 69.208745   203.138.129.30   9.1.1.2      SSH       294 Client: Encrypted packet (len=240)
533 69.227759   146.65.96.8      9.1.1.2      SSH       294 Client: Encrypted packet (len=240)
534 69.245270   62.65.42.243     9.1.1.2      SSH       294 Client: Encrypted packet (len=240)
535 69.257280   8.186.84.251     9.1.1.2      SSH       294 Client: Encrypted packet (len=240)
536 69.272289   17.33.244.197    9.1.1.2      SSH       294 Client: Encrypted packet (len=240)
537 69.296308   212.225.66.250   9.1.1.2      SSH       294 Client: Encrypted packet (len=240)
538 69.319824   168.86.83.103    9.1.1.2      SSH       294 Client: Encrypted packet (len=240)
539 69.333834   244.155.20.20    9.1.1.2      SSH       294 Client: Encrypted packet (len=240)
294 bytes captured (2352 bits) on interface 0
```

```
┌─[root@parrot]─[/home/ptc
└──╼ #hping3 --flood --rand-source -S -c 10000 -w 32 -d 240 9.1.1.2 -p 22
HPING 9.1.1.2 (eth1 9.1.1.2): S set, 40 headers + 240 data bytes
hping in flood mode, no replies will be shown
```

```
root@ptc-VirtualBox:~# ssh root@9.1.1.2
ssh: connect to host 9.1.1.2 port 22: Connection timed out
root@ptc-VirtualBox:~# ssh root@9.1.1.2
ssh: connect to host 9.1.1.2 port 22: No route to host
root@ptc-VirtualBox:~#
```

Figure 12: An attacker performing a protocol DDoS attack (Ballal, 2018)



```
root@ubuntu:~# hping3 -c 10000 --flood -S -w 64 -d 240 -p 443 --rand-so
urce [redacted]
HPING [redacted] ([redacted] [redacted]): S set, 40 headers + 240 dat
a bytes
hping in flood mode, no replies will be shown
```

```
root@ubuntu:~# curl -v https://[redacted]
* Rebuilt URL to: https://[redacted]
*   Trying [redacted]...
* TCP_NODELAY set
* connect to [redacted] port 443 failed: Connection refused
* Failed to connect to [redacted] port 443: Connection refused
* Closing connection 0
curl: (7) Failed to connect to [redacted] port 443: Connection ref
sed
```

Figure 13: A web application after a TCP SYNC flood attack (Ballal, 2018)

In these examples, the DDoS occurs when the attacker sends multiple SYN requests from multiple spoofed Internet Protocol (IP) addresses to disrupt the service.

Table 6: Type of DDoS attacks with symptoms and impacts

DDoS attack					
Type	**Characteristics**	**Scope**	**Challenge**	**Symptoms**	**Impacts**
Volumetric attacks	Generate a massive traffic volume to saturate the bandwidth and create traffic congestion	Focus on the bandwidth saturation	Reduce the attacks that combine reflection and amplification threats with botnets	• Website or service becomes significantly slow or suddenly unavailable. • Increases the volume of network traffic, spam emails either from one IP address or multiple IP addresses. • Inbound traffic overflow from users with similar	• Website breakdown could negatively influence the search ranking • The servers or websites are more vulnerable
Protocol-based attacks	Over-consumption of server resources and network intermediaries (firewall, load balancer, etc)	Focus on resource depletion	Weaknesses of internet protocols (TCP, UDP, ICMP, etc…)		

Application Layer attacks	Exploit weakness in the application layer in order to take out applications, online services, or websites	Require less amount of bandwidth than the other attacks.	Challenge the distinction between legitimate and malicious traffic in distinguishing legitimate traffic from malicious.	behavior patterns (device type, location, or web browser).	and susceptible to further attacks, such as hacking or data theft.

To summarize, over the year and especially with the Covid 19 pandemic cyberattacks becoming more and more sophisticated, widespread, targeted, and undetected. People work at home, so they are more vulnerable to all types of cyberattacks. People and organizations face more and more targeted ransomware, the new variety of phishing attacks, the evolution of mobile malware attacks, the rise of cyber insurance, and risky business with IoT devices.

With technological innovation, some environments are still not immune to cybersecurity attacks. There are easily vulnerable to attacks like DNS attacks, Man in the middle, Ransomware attacks, Phishing, …etc which remain the most relevant and need advanced methods of protection against them (*Mohammed et al., 2018*). Security experts are always updating their strategies against cyberattacks to provide organizations and individuals the best protection levels. There is a need to teach and implement the good practices related to internet security, which use diverse and complex passwords, backup data, updated programs, and encryption.

2.3.5 Comparison between cybersecurity attacks

Overall, with the covid19 pandemic users are facing different types of attacks as mentioned in this work. There are various attacks in cyberspace and the most common attacks are fraud, malware, spam, phishing, DDoS, ransomware, disabling firewall and antivirus, logging of keystrokes, and malicious URL. Some statistics show (brooks, 2021):

- Data breaches mostly occur in hospitals, in fact, more than 90 percent of all healthcare organizations reported at least one security breach in the last three years.

- The US Federal Trade Commission received 1.4 million reports of identity theft by 2020, a double increase as compared with 2019.

- Among the hundreds of millions of attempted cyberattacks that occurred every day throughout 2020 showing malware increased by 358% overall and ransomware increased by 435%.

- Over 80% of reported cyber attacks are Phishing attacks. For example, Google has registered 2,145,013 phishing sites as of Jan 17, 2021, compared to 1,690,000 on Jan 19, 2020 (up 27% over 12 months).

- There is at least one ransomware Victim every 10 Seconds in 2020. For example, ransomware attacks 1 in 5 Americans.

- Netscout Threat Intelligence register over 4.83 million DDoS attacks in 1H 2020. That means, there are 26,000 attacks a day or 18 attacks per minute.

Given the statistic shown by the different attacks, we can say that the most evasive attack categories are Phishing, malware, and DDoS because they are considered critical attacks on cyberspace. We found a clear correlation between the increase of malware sites and phishing sites during the pandemic. Indeed, individuals and organizations received more and more sophisticated phishing emails due to several factors, including:

- Lack of security assistance,
- System vulnerability
- Bad cybersecurity hygiene

The authors (Hadnagy & Fincher, 2015) highlight some cognitive aspects that affect the user's reasoning and favor cyber attacks. According to (*Aïmeur et al., 2013)* the attacker analyzes the victim's behaviour, and picks the best profile by which he can manipulate the victim. For instance, by mimicking one of the victim's behaviour or friends, the attacker is more likely to attack him into downloading malicious software or clicking on a malicious link.

Moreover, the authors (*Ferreira et al., 2015*) identify the principles of persuasion used by the attacker that influence the user behaviour. In fact, phishing emails use some tricks including the use of boldface, different text colors, repetition of information, … etc to manipulate and convince victims of the authenticity of the emails. As we can see, phishing attacks use more persuasion strategies.

Finally, depending on the context, most of the time attackers hide their attacks on forms of usual interactions with their victims such as attachment files, birthday notes, videos links, …etc. the goal of this technique is to distract the victim, so the attackers can easily encourage them to reveal sensitive information, download or click on malicious links.

For this reason, we consider social engineering as the most harmful attack (Beckers & Pape, 2016). In fact, social engineering attacks exploit the weakest layer of people's vulnerabilities, so any person can be the victim. This attack uses some psychological techniques dividing into three: understanding the targeted victim, developing a perfect plan, and launching that plan (Corradini, 2020).

During our literature review of cyber attacks, we found that each attack is unique, with five dimensions: the goal of the attack, the source of the attack, the target, the attack vector, and the impact. A cyber attack’s goal and source influence the methods employed. The target can be considered at a very high level, such as the targeted organization type, or a lower level such as the targeted individual type within the computing system or network. The attack vector considers the path by which the attack is carried out.

The major links between these dimensions are vulnerabilities. In fact, each target has a big amount of vulnerabilities, and, each attack vector is suited to exploit these vulnerabilities.

A lot of research focuses on the impact of the cyberattack at a different level, including the hardware level (computing system and network), or the information level (corruption, disclosure, theft, or compromise of information). It also proves that the impact may take the form of another attack vector.

As new types of attacks emerge, and traditional cyber attacks become more sophisticated, future research plans aim to understand and address what are the operational outcomes across technological, user-centered, and socio-organizational variables.

2.4. Natural Language Processing

In today's fast-moving world, it is hard to imagine an intelligent system without natural language processing. Natural language processing appeared for the first time with the term "translating machine" in the mid-1930s, one of the precursors of this technology was Georges Artsrouni (Hutchins, 1997), who developed a bilingual dictionary that can map the words of one language to another. However, this approach was incomplete, and the second precursor Peter Troyanskii (*Johri et al., 2020*) proposes an approach to strategy to tackle the grammar for a language in the translating machine using the electromechanical technology of the 1930s and 1940s.

During World War II, the first use of Natural language processing was done by the German through the machine named "Enigma" (Prasad & Kumari, 2020) used for encrypting the secret message of the Germans into the secret code. Over the years, many researchers have contributed to the NLP field and the development of the current approaches. The algorithms were based on handwritten rules (Kanakaraddi & Nandyal, 2018). Later, in the 1980s, Machine Learning (ML) emerged as the most significant change from handwritten rules to concept making. ML algorithms provide an effective way to analyze and interpret sample data to make predictions or decisions. Currently, the other biggest innovation in the NLP field is the integration of Deep Learning (DL). In fact, DL can solve some of the most ambiguous language problems.

NLP has been put to use in the development of many intelligent systems like Chatbots *(Ayanouz et al., 2020*), recommendation systems, opinion mining (*Sun et al., 2017*), sentiment analysis, intelligent assistants…etc. With the emergence of various question answering recommendation systems and the progress of NLP, personalized Q/A recommendation systems achieved good results.

2.4.1 Natural Language Language Tasks

Multiple NLP tasks must be done in ways that help the computer to understand what it's ingesting. Some of these tasks include the following:

a. Speech recognition

Speech recognition is an NLP task that consists of converting voice data into text data *(Ravanelli et al., 2019)*. The main goal of this task is to be able to “hear”, “understand”, and “act upon” voice data. In the early 1950s, Davis, Biddulph, and Balashek developed the first speech recognition system: Automatic recognition of spoken digits for a single speaker (*Davis et al., 1952*). The role of this system was to analyze, extract and recognize information about the speaker's utterance.

b. Part of speech tagging

Part-of-speech tagging (PoS tagging) is a process that consists in adding a part-of-speech category to each token within a text. Common PoS tags are *verb*, *adjective*, *noun*, *pronoun*, *conjunction*, *preposition*, *intersection*, among others. Figure 14 shows an example of POS tagging.

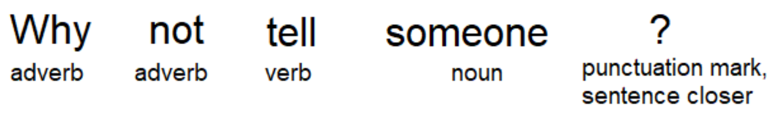


Figure 14: Example of POS tagging

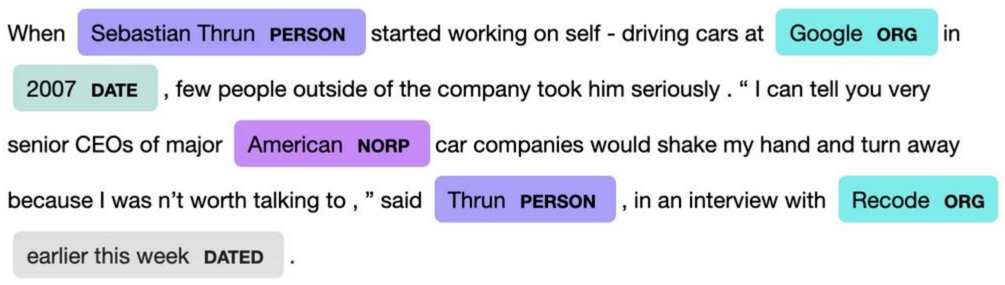
PoS tagging technique is used to classify words and, therefore, parse the structure of sentences. The next task can also be considered a challenge and a potential limitation of NLP in its current state of the art and that is how to handle inherent ambiguity in languages.

c. Word sense disambiguation

Word sense disambiguation (WSD) (*Y. Wang et al., 2020*) is the NLP technique that determines the words' sense depending on their context. Word sense disambiguation can be divided into two main techniques: knowledge-based and supervised approach (Navigli, 2009).

d. Name entity recognition

Named entity recognition (NER) (*Luo et al., 2020*) is a technique that involves extracting entities from within a text that refers to names, places, organizations, email addresses, and more. NER is an important NLP task that can be divided into two: the identification of proper names in text and the classification of these names into a set of predefined categories of interest. This technique can be applied in many applications of NLP such as machine translation, internet search engines, automatic indexing of documents, automatic question-answering, information retrieval, etc.



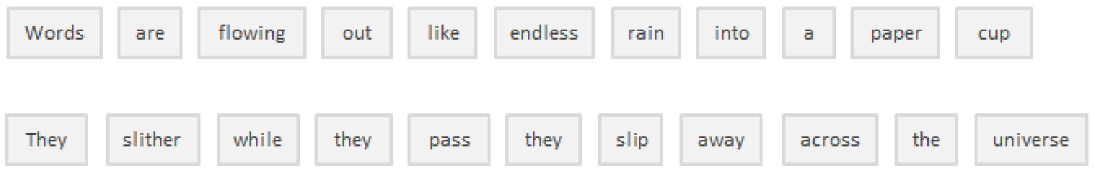
When Sebastian Thrun PERSON started working on self - driving cars at Google ORG in 2007 DATE , few people outside of the company took him seriously . " I can tell you very senior CEOs of major American NORP car companies would shake my hand and turn away because I was n't worth talking to , " said Thrun PERSON , in an interview with Recode ORG earlier this week DATED .

Figure 15: Example of Named Entity Recognition (Prakash, 2020)

e. Tokenization

Tokenization is a fundamental task in natural language processing that segmented words into semantically useful units called tokens. Sentence tokenization breaks a string up into sentences within a text, and word tokenization breaks a string up into words within a sentence. Generally, word tokens are separated by blank spaces, and sentence tokens by stops. It's possible to perform complex tokenization for more high-level structures, like words that often go together, otherwise known as collocations.

For the sentence: "Words are flowing out like endless rain into a paper cup," and "They shilter while they pass, they slip away across the universe", the result of tokenization would be:



Words	are	flowing	out	like	endless	rain	into	a	paper	cup	
They	slither	while	they	pass	they	slip	away	across	the	universe	

Figure 16: Example of Tokenisation (Yse, 2019)

Next, is the task that usually follows Tokenisation in the chronological order of text processing.

f. Lemmatization & Stemming

Lemmatization is a process that strips words to their base form and groups them together in different forms of the same word. Lemmatization transforms words to their dictionary form called a *lemma*. For example, verbs in the past tense are changed into the present (e.g. “arrived” is changed to “arrive”) and synonyms are unified (e.g. “worst” is changed to “bad”). Each word has a Part of Speech tag attached to it before lemmatization occurs by using a WordNet dictionary.

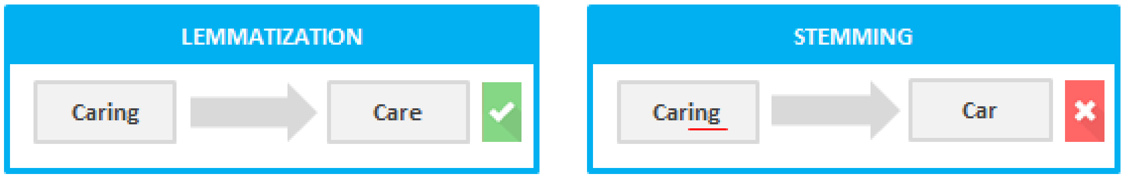


Figure 17: Example of Lemmatization (Yse, 2019)

Stemming is a process used to reduce a word to its stem or root form and it is used in many areas of computational linguistics and information-retrieval work.

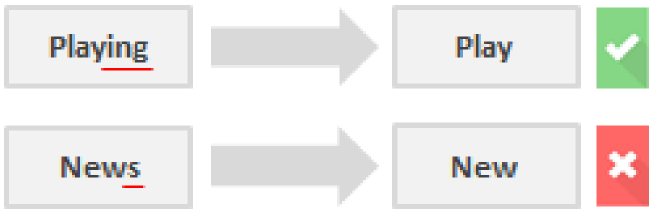


Figure 18: Example of stemming (Yse, 2019)

At this stage, the base form is obtained but one more step is needed to conclude the basic NLP tasks.

g. Stopwords removal

Stop words are the most common words used in a particular language. The examples of English stop words include pronouns, preposition, adverb, … “the”, “is”, “my”, “on”, …etc. These words can generally be removed without changing the value or the sense of the corpus during the NLP. The stopwords can be pre-selected or customized. A general approach is to adopt pre-defined stop words before adding words to the list. These words are usually filtered out before some Natural Language Processing tasks such as Topic Modelling and Text Classification can occur using machine learning models. However, stop word removal is not important anymore with deep neural networks. In fact, natural language processing tasks in deep learning aim to learn how the language works from the dataset as a sequence and how each word is connected to its neighboring words. For that reason, removing stop words can affect contextual information. Nevertheless, in this project, we tried a machine learning algorithm, and, we proceeded with stop words removal.

2.4.2 Natural Language Processing Algorithms

There are two commonly used natural language processing algorithms: rule-based algorithm and machine learning-based algorithm.

a. Ruled-based NLP algorithm

The ruled-based NLP algorithm is one of the oldest NLP approaches largely used between 1950-the 1990s. The rule-based NLP algorithm contains two components: a declarative component that represents the linguistic knowledge which includes the grammar and the lexicon of the language, while a procedural component corresponding to the analysis strategy which specifies in detail each of the operations involved in the process of analysis.

b. Machine learning NLP algorithm

Machine-learning approaches include probabilistic modeling, likelihood maximization, and linear classifiers. Unlike rule-based, Machine learning NLP involves a set of statistical techniques for identifying parts of speech, entities, sentiment, and other aspects of the text. The algorithm uses a statistical method, where the system analyzes the training set (annotated corpus) to build its knowledge, rules, and classifiers.

Table 7: The table below sums up all the strengths and weaknesses of both approaches

	Advantages	**Disadvantages**
Rule-based	• Declarative • Easy to understand the language phenomenon • Doesn't require a massive training corpus • Easy to maintain and debug • Easy to incorporate domain knowledge	• Heuristic • Requires skilled developers and linguists
Machine learning-based	• Trainable • Adaptable • Fast development • Easy to scale	• Requires training corpus with label • Requires retraining for domain adaptation • Difficult to debug • Requires ML expertise to use or maintain

This section dives deep into natural language processing, the NLP tasks, and the different NLP algorithms. We highlighted some NLP tasks such as Tokenization, stopwords, lemmatization, …, etc.

In summary, upon beginning this chapter, it was shown that the value of recommendation systems and personalized recommendation systems were evident in both academia and industry. A lot of researchers have been making contributions to ensure that everybody (business, individual, and organization) may positively benefit from this kind of system to increase the user experience, in terms of revenue, customer, and performance throughput. This chapter discussed various real-world existing personalized recommendation systems which used the different three recommendation approaches. Also, discuss their strength as well as limitation in the time. Chapter 2 presents some cyber attacks experienced by the individual and organization in this Covid 19 pandemic. The most common are: Malware, DDoS, Phishing, and Ransomware. Finally, it introduces Natural Language Processing as an important part of the development of a recommendation system for a better understanding of the user intent or preference. All the information presented in this chapter will be used to build the "personalized question-based cybersecurity recommendation system" in the next section.

Chapter 3 – The Proposed Personalized Question-based Cybersecurity Recommendation System

We realized our Cyberhelper, a personalized recommender system that allows each user to ask, get useful information, and solve their internet issues or cyberattacks. Moreover, the proposed system overcomes the recurrent problem associated with the traditional Q/A recommendation systems, where at the first interaction, the system asks users to express their preferences (Recommendation, 2021) and makes assumptions on user goals, interests, and preferences based on their historical activity.

In fact, the Cyberhelper constantly improves its “understanding” of the user thanks to its user modeling. In fact, user interest and preferences over time, user background, knowledge, etc. are all important parts of customized cybersecurity recommendations.

Since the Cyberhelper is targeting people with good, little, or no cybersecurity background, it takes into consideration the specific needs of everybody. In fact, the intents (Cai, 2020) behind each user utterance are varied and might refer to different preferences such as “Ask for recommendation”; “ Get information”; “ Give feedback”, and other sub-intents. Therefore, we apply these preferences in the proposed system.

In the following sections, we detail the methodology for the proposed architecture of the platform.

3.1 Cyberhelper Methodology

Cyberhelper, the personalized recommendation system for personalized cybersecurity assistance is not only based on user preferences but also on the current context and user model to give a real-time experience. This system continuously interacts with the users and analyses their ratings to update and optimize the recommendation results. The main goal is to recommend the best cybersecurity solutions that best fit users in their specific current context.

Cyberhelper aims to improve the user’s safety behavior and cybersecurity culture. It helps the users understanding the impact of the cyberattacks, how to be prepared against a cyberattack, and what makes them vulnerable in order to assume their role in the security process and make sure that they will react appropriately when facing a real cyberattack. To do so, Cyberhelper is capable not only to answer the user’s query but also to take into consideration similar users and learning, progressively, the user's preferences and their historical interactions with the system. Throughout this section, the description of Cyberhelper is detailed.

3.1.1 Problem Description

A successful personalized question-based recommender system requires a good interaction between the system and the user.

In this work, our objective is to come up with a personalized recommender system that is a cybersecurity assistant. The three keywords in this objective are “personalized”, “recommender system” and “cybersecurity” and this brings this section to the following questions which can also be seen as the challenges to overcome or this research’s axes:

- Which recommendation techniques/algorithms to use for the recommender system?
- How to achieve personalization? This ties with the previous bullet point but also with another point which is the user and context model.
- How to represent the data on which recommendations are based? This connects to the “cybersecurity” keyword. The same recommender algorithm can be used for a multitude of applications provided that a proper database serves as its input.

Li et al. (2017) have shown that a good and effective recommendation system can increase users’ trust in the system. Moreover, Zhang et al. (2021) explain that a good personalized recommendation system should have transparent system logic and provides well-detailed information about recommended items.

This research proposes a Cyberhelper based on a hybrid algorithm of multiple recommendation algorithms. This choice is made based on the fact that this research is based on two main parts:

Q/A recommenders and personalized recommender systems. Hence the hybrid approach is composed of three main different algorithms, as shown in Figure 19:

- Knowledge-based algorithm: Question/Answer interactions.
- Collaborative-based filtering algorithm: Similarity between users.
- Content-based filtering algorithm: Incremental learning of user behavior and preferences.

The first input is the user information, which describes their Age, Education level, Work domain, and Position. Also, the context information, which represents the circumstances of using the system such as Time, Day, Browser, Location, and Device. After collecting that information, the user will enter their query.

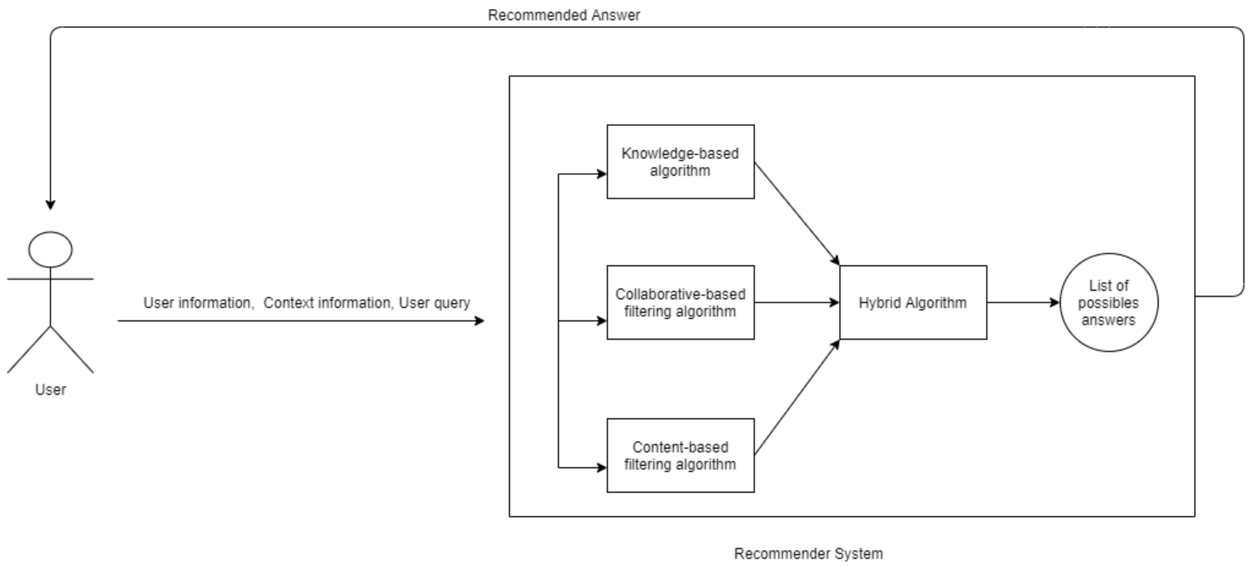

Recommended Answer
User
User information, Context information, User query
Knowledge-based algorithm
Collaborative-based filtering algorithm
Content-based filtering algorithm
Hybrid Algorithm
List of possibles answers
Recommender System

Figure 19: Global architecture of our proposed algorithm

The system works according to these three modules, its objectives are that the internal process helps the user to get a good answer to their problem, and makes their list of possible answers. The Cyberhelper output is the highest-rated answer by the user among all the possible answers generated by the different algorithms. The details of our algorithm are discussed in the next section.

3.1.2 Problem formulation

This section presents the internal functioning of each algorithm of the global architecture of the proposed system. The detailed architecture is given in the flowchart.

In the beginning, the user presents their characteristics: age, Education level, Work domain, and Position, as well as the current context information: Time, Day, Browser, Location, and Device. Then, the user formulates their problem in natural language, and the Cyberhelper proceeds with some preprocessing techniques in order to extract the keywords of each category. Using those keywords, the system predicts the category of the user input.

Once the attack category is determined, the different algorithms: the knowledge-based algorithm, the collaborative-based filtering algorithm, and the content-based filtering algorithm are applied separately.

The whole process is combined in the hybrid algorithm-based Cyberhelper and the results of each algorithm mentioned will be weighed to find the appropriate recommendation results.

a. The solution Flowchart

This part presents the flowchart of our algorithm. In order to provide a personalized recommendation system, Cyberhelper is built on a hybrid algorithm that combines the knowledge-based algorithm, the collaborative-based filtering algorithm, and the content-based filtering algorithm is implemented.

The process through which recommendations are given to the user is detailed in Figure 20 that outlines the system's flowchart. The first part is to preprocess the user information and queries while retaining useful information and predicting the category of the attacks. The second phase is based on the attack prediction, obtains various recommendation results through the three algorithms: Knowledge-based, collaborative-based filtering, and content-based filtering algorithm. The third phase calculates the weights for each result. The fourth phase is to provide the final recommendation results by combining the results of the three recommendation algorithms. The last phase is to show the personalized recommendation results to the user through an interactive interface and record the user's rating information to update the recommendation results.

Flowchart explanation:

This subsection is dedicated to detailing the different steps taken by Cyberhelper based on the hybrid algorithm that will be further detailed in the following sections:

- The user initializes the process by logging in to the system and typing their input

- The user input/query is preprocessed and the category of the attack is predicted. Table 19 shows an example of Category-Keyword matching from the dataset, where we have a specific list of keywords for each category. To predict the corresponding category of the user input, we compare the presented words in the input with each category list of keywords, see Algorithm 1.

- Based on the previous prediction, the hybrid algorithm (Knowledge-based, collaborative-based filtering, and content-based filtering) is applied as follows:

    - In the Knowledge-based Algorithm, we first have the general questions and answers dataset where for each question and answer scenario, there is a specific recommendation.
    - In the Collaborative-based filtering Algorithm, there is the user information such as (age, id, Education level, …) and also the context information (time, day, device, …). The system computes its similarity with the other user and provides a recommendation based on the other similar users.
    - In the Content-based filtering Algorithm, there is the user Information such as (age, id, Education level …), and context information (time, day, device …) and for each scenario (user information + context) a specific answer.

- After that, the system trains an incremental Adaptive random forest machine learning algorithm that will understand the distribution of users and context information in a way that will be capable of predicting the right recommendation for each user.

- In the end, the system will return three recommendations, each algorithm will return one possible response and the user will choose the recommendation that fulfills their needs. At this point, the recommendation is done. It is important to highlight that our system is using a ranking mechanism that will produce the most convenient solutions for a specific attack. So that, in the end, we will have three recommendations that can resolve the problem. As a user satisfaction measure, a rating is given to each of the recommendations. The system will store the highest-rated recommendation for future purposes, in case the user encounters similar problems.

The proposed system does not suggest recommendations to the user and then provides the one that the user rates higher. The rating is a measure of satisfaction that comes after the recommendation. It serves to better personalize future recommendations by letting the system know what the user found to be the most convenient. I added an example in the appendix of the present document to showcase the recommendation process followed by the user rating.

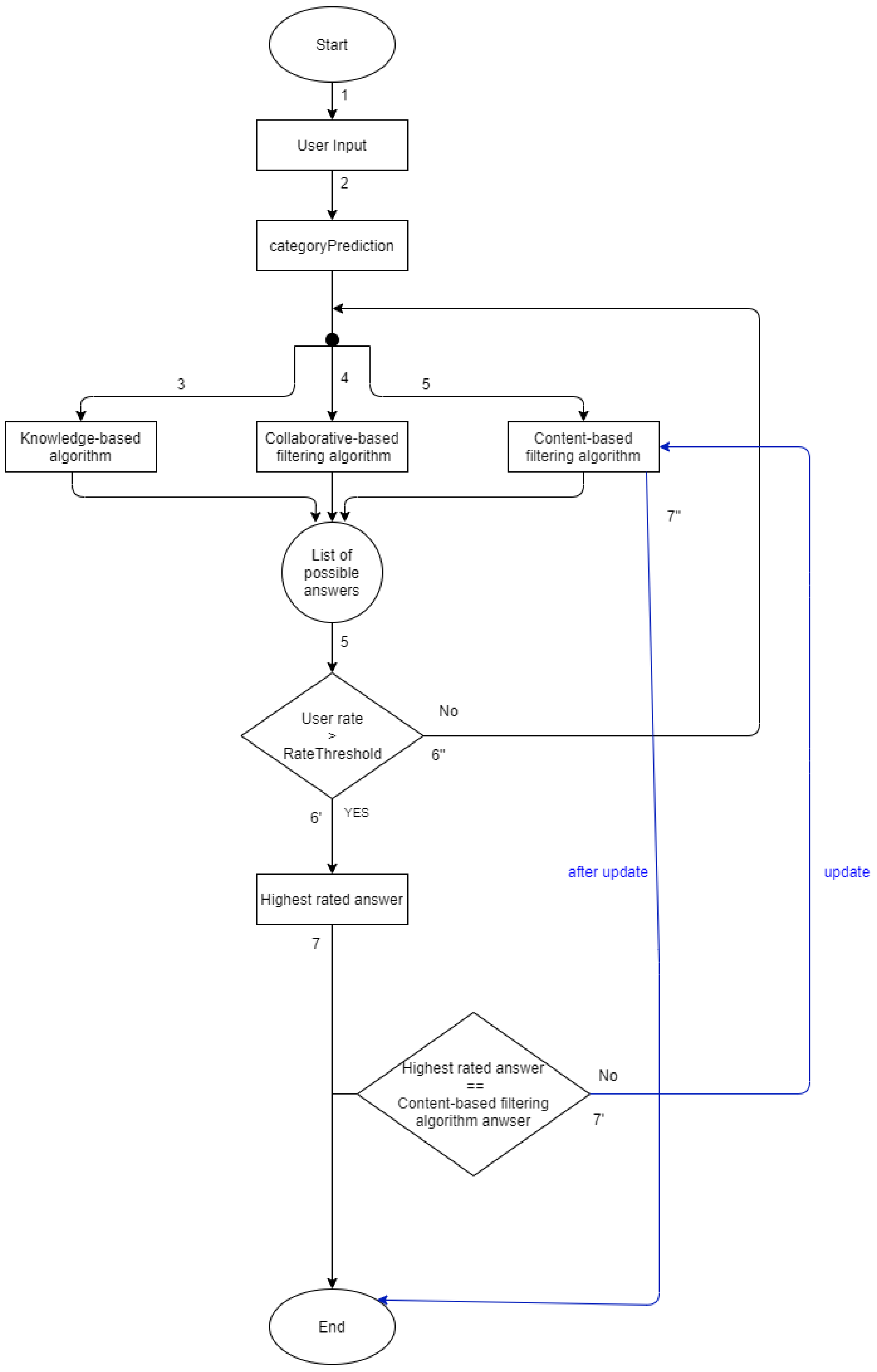

Start
1
User Input
2
categoryPrediction
3
4
5
Knowledge-based algorithm
Collaborative-based filtering algorithm
Content-based filtering algorithm
7"
List of possible answers
5
User rate > RateThreshold
No
6"
6'
YES
Highest rated answer
after update
update
7
Highest rated answer == Content-based filtering algorithm anwser
No
7'
End

Figure 20: Hybrid algorithm flowchart

Obtaining the recommendation list requires specific elements: User model, context, user query, and recommendation algorithms. The system mines users' interest information from multiple dimensions, precisely user model and context. Meanwhile, the system processes the knowledge-based data and gets the relation between the cybersecurity solutions. According to the needs of each recommendation algorithm, the system handles related information as the basis for subsequent analysis.

The Collaborative-based filtering algorithm is based on a group of users with the same problems. CF uses the similarity between users, then, it combines the user's historical information to obtain the recommendation result.

3.2 Platform Architecture

This section introduces the architecture of the system, which consists of eight essential modules. As shown in Figure 21, the modules are as follows: the user interface, the request preprocessing module, the user modeling module, the Recsys modeling module, the domain knowledge, and the recommendation module. The user interface serves as the medium through which the user can easily interact with the different modules of the system. The request preprocessing module analyzes the raw text input by the user and determines the exact data useful and understandable by the system. The user modeling module contains some information related to the user such as the user model, the context, the ratings, … etc. The Recsys modeling module executes the proposed algorithm. The domain base contains cybersecurity information divided by threat, solution, and service. The recommendation module generates a list and sends the personalized recommendation to the user. The hybrid algorithm associated with three different algorithms is the method that we have adopted to provide the personalized recommendation.

Figure 21 shows the basic architecture of the personalized question-based cybersecurity recommendation system, while Figure 22 shows the detailed architecture of the proposed system.

The following sections discuss each of the elements mentioned above and their functions.

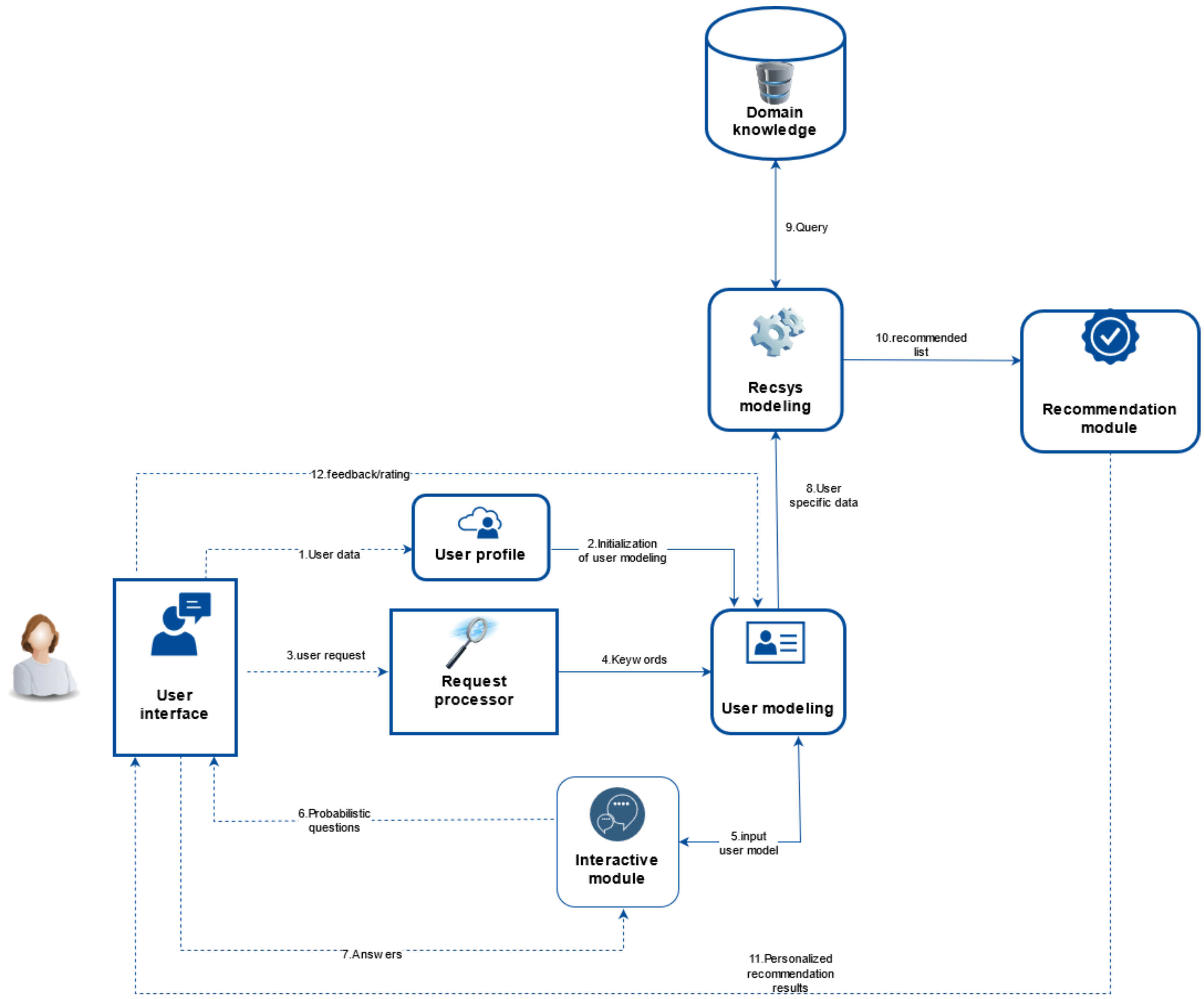


Figure 21: Architecture of Personalized Question-based Cybersecurity Recommendation System

After presenting the global architecture of the Cyberhelper, figure 22 shows the detailed architecture of each module. It presents the fundamental organization of the system as a whole along with its modules, sub-modules, and their inputs and outputs.

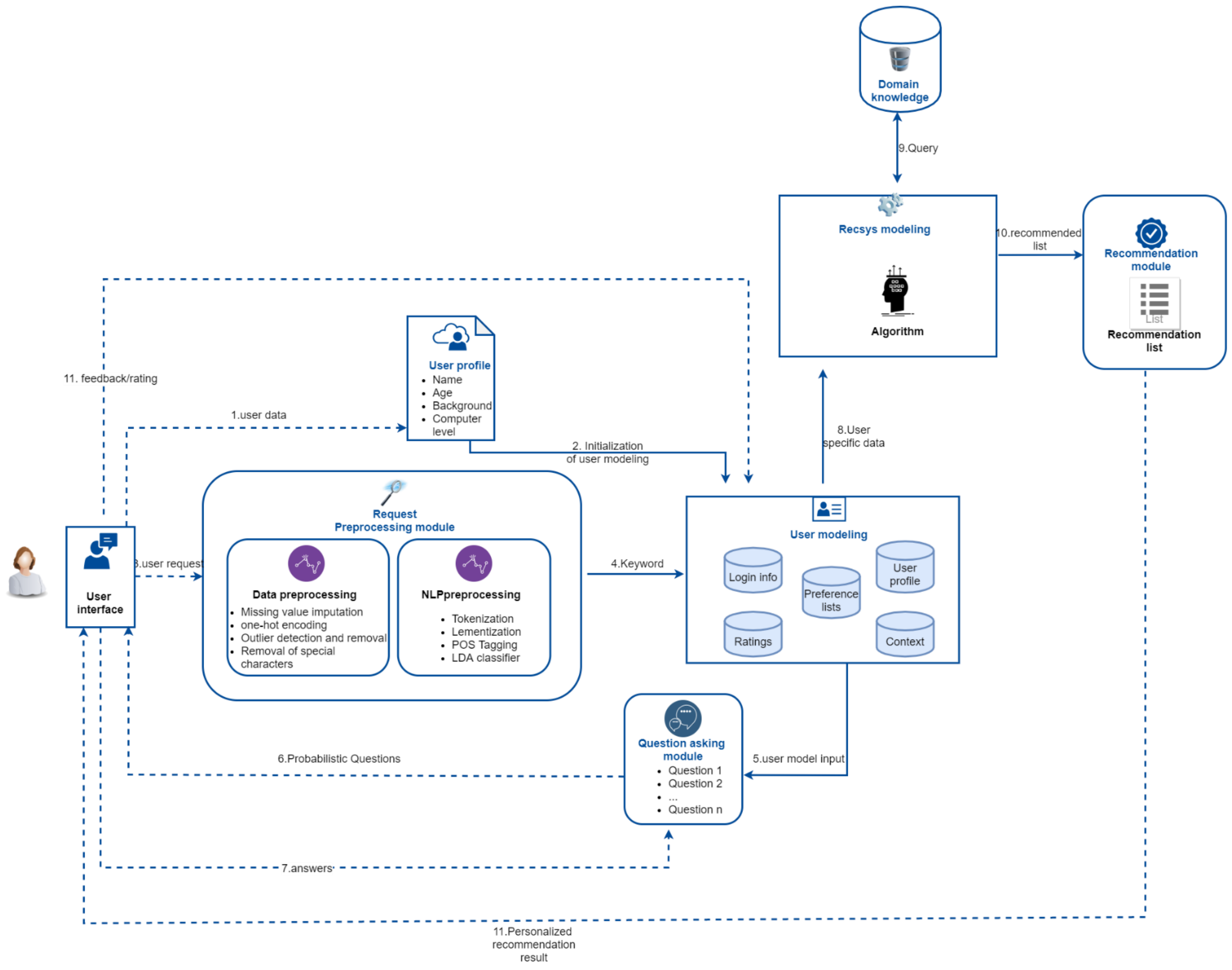


Figure 22: Detailed architecture of the Personalized Question-based Cybersecurity recommendation

The detailed architecture focuses on Cyberhelper modules and their sub-modules developed.

3.2.1 The query preprocessing

The Cyberhelper executes a preprocessing process that allows the user and the system to communicate in a language that both can understand. In fact, the preprocessing process is one of the most important parts of the implementation of our system, which serves to process the user text input into something understandable by the user modeling module. Meanwhile, one of the current issues experienced by the traditional preprocessing techniques is the data quality which is very important to increase the accuracy of the system. Usually, the preprocessing process (Guan, 2018) is done by the following two main processes: text preprocessing and NLP preprocessing. Further details are included in Figure 23.

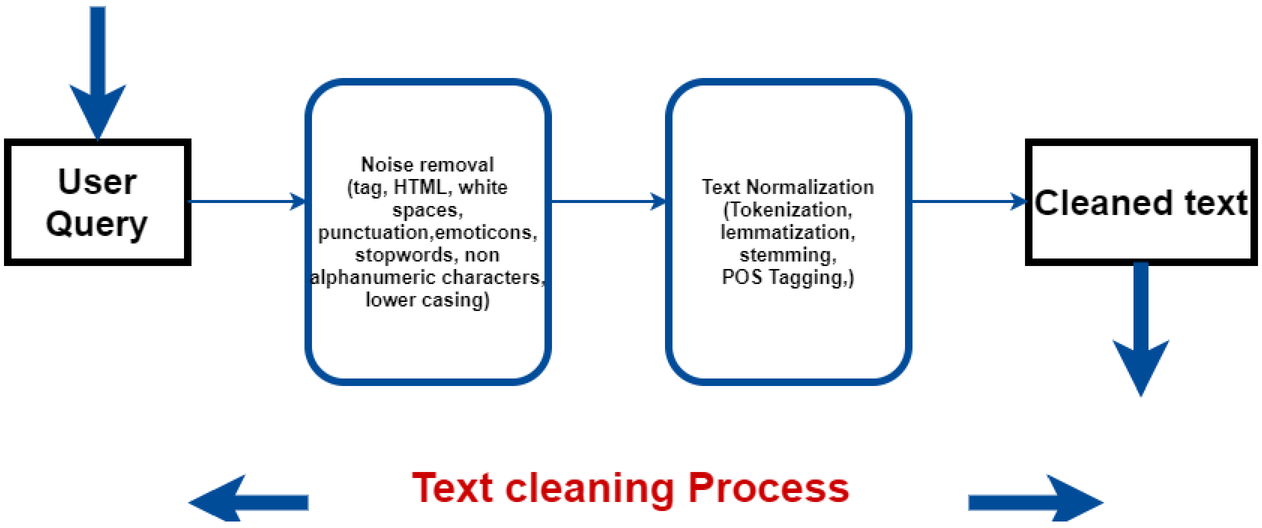

User Query
Noise removal (tag, HTML, white spaces, punctuation,emoticons, stopwords, non alphanumeric characters, lower casing)
Text Normalization (Tokenization, lemmatization, stemming, POS Tagging,)
Cleaned text
Text cleaning Process

Figure 23: The text cleaning process

Text preprocessing consists of filling in the missing values, remove outliers, errors, noise, special characters or icons, and even irrelevant information (Hariharakrishnan, 2017). The other process is the NLP preprocessing (Solangi, 2018) refers to two main topics: natural language understanding and natural language generation. In the first one, the system tries to read, understand and interpret the text. For the second one, the system is trained to learn and to generate realistic text like a human. Previous works have shown some limits about the merits of the NLP preprocessing methods, because of their harmful effects which include the exclusion of some minority communities due to demographic bias (Mieskes, 2017), the risk of bias in some systems is due to overgeneralization of model predictions. Moreover, some researchers are more concerned about privacy and intrusion in people's lives when collecting data. It’s become very challenging to create machine learning models that take into consideration privacy (Ribeiro-Navarrete, 2021), confidentiality, integrity, and authentication of the user.

To address the above-mentioned problems, many studies focus now on the way to build intelligent systems that can prevent users from disclosing information. In the Cyberhelper’s architecture, the two main preprocessing methods: Text preprocessing and NLP preprocessing have been used. *Text preprocessing* consists of cleaning the data by filling the missing value, removing noise, outlier, …etc. Using these factors increases the quality of our data. NLP preprocessing consists of understanding and extracting the meaning of the text, considered essential to improve performance and the accuracy of the system. So, in this module, the procedures are divided as follows: the text preprocessing and the NLP preprocessing as shown in Figure 24.

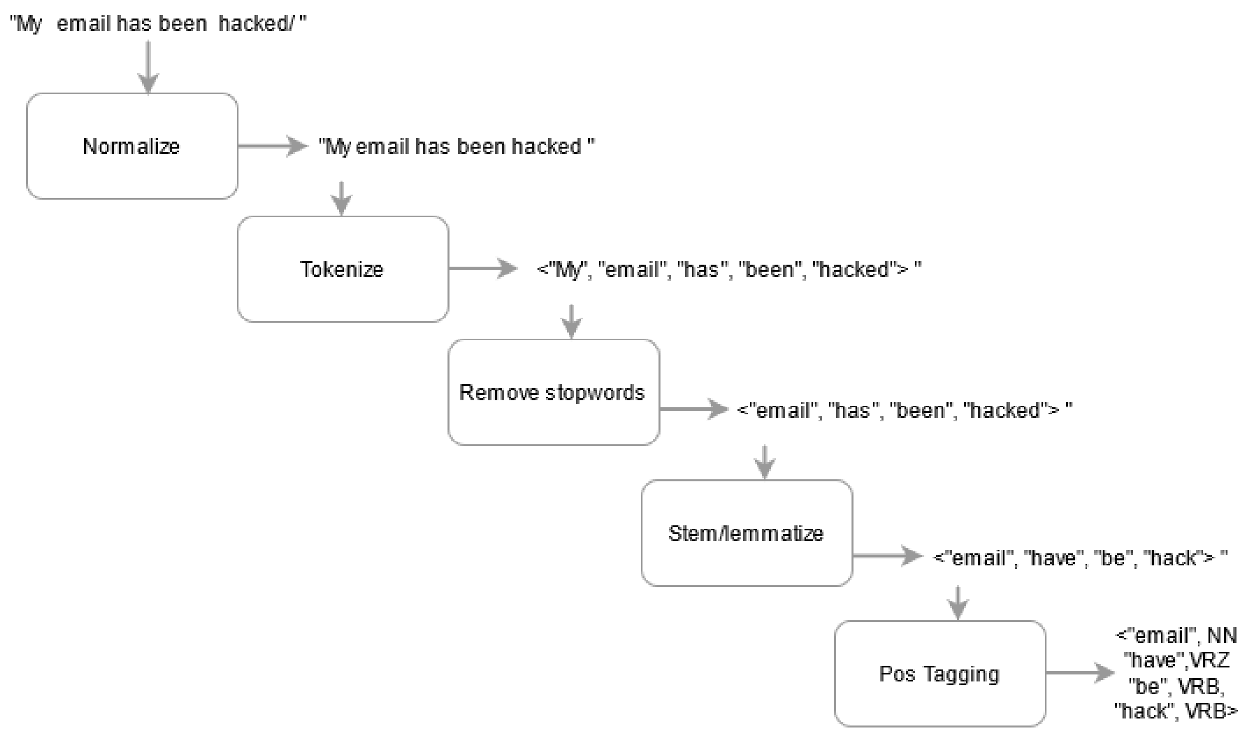

"My email has been hacked/ "
Normalize
"My email has been hacked "
Tokenize
<"My", "email", "has", "been", "hacked"> "
Remove stopwords
<"email", "has", "been", "hacked"> "
Stem/lemmatize
<"email", "have", "be", "hack"> "
Pos Tagging
<"email", NN
"have",VRZ
"be", VRB,
"hack", VRB>

Figure 24: Example of NLP preprocessing step

The following sections discuss these different stages in detail.

a. **The Text Preprocessing**

Text preprocessing is the first step of our preprocessing process which consists of cleaning the data. Therefore, this technique is based only on the unstructured text that usually contains some noises, errors, outliers, …, etc. Sometimes the user input contains unwanted noise such as spaces, special characters, numbers, and HTML formatting that need to be removed. Some algorithms are very sensitive to the presence of outliers and noise contained in the data, which affect the classification performance by creating for example the wrong nearest neighbors. To avoid this kind of constraint, the use of a well-known text preprocessing technique: noise filter is required. In (Rekha, 2018), the authors introduce a "noise filter with Adaptive Boosting Algorithm (AdaBoost)" for effective classification. As mentioned in the description, besides the presence of noisy values, it's also possible to find the data values called outliers (Fearnhead, 2019). The author Angiulli (2014), the authors classify the approaches for outliers and noise detection into 3 categories: supervised, semi-supervised, and unsupervised methods. In fact, the supervised methods use data labeled as normal or abnormal contrary to semi-supervised methods which use only data labeled normal. The unsupervised methods don't need labeled data to search for outliers.

Another important issue in text preprocessing is to fill the missing values. In most cases, the dataset contains incomplete value or information that make the text difficult to understand. To address the above-mentioned problem, some studies use the missing value imputation method (Tsai, 2018), which consists to replace the missing value from a chosen set of observed data using some statistical methods, such as k nearest neighbor imputation (KNNI) (Pan, 2015), or support vector machine (SVM).

By conducting a survey of the review of text preprocessing, the author (Hickman, 2020) described and classified each text preprocessing used in the data mining field, also we can easily understand the impact of text preprocessing on the result of our machine learning algorithm.

b. **The NLP Preprocessing**

The preprocessing allows the user and the system to communicate in a language that both can understand. In fact, the preprocessing process is one of the most important parts of the implementation of our system, which serves to process the user text input into something understandable by the other modules.

Based on the NLP preprocessing, the system should be able to analyze the user query, extract meaningful keywords and build a strong classifier. Analyzing user queries involves 1) breaking a sentence into tokens (tokenization); 2) lemmatizing each token (lemmatization), and 3) removing each useless word (stopwords removal).

1. **Tokenization**

The authors (Rai & Borah, 2021) described tokenization as the mechanism of splitting a sentence or document into chunks called a token. Tokenization is a fundamental phase of every Natural Language Processing (NLP) implementation, but some additional information should be taken into consideration such as the named entities.

For example, segmenting on spaces converts the sentence: *What should I do if I suspect a cyberattack?* into the list of tokens [*What, should, I, do, if, I, suspect, a, cyberattack*].

Many tokenization tools such as Stanford Tokenizer[1], OpenNLP Tokenizer[2], … are available on programming languages, including C, R, Python, Ruby, Java, and Clojure have libraries that can tokenize English words.

This system uses Python's NLTK library[3] which has many tokenizers including Word Tokenizers and Regular Expression Tokenizers. Word Tokenizers segment a string into words whereas Regular Expression Tokenizers splits a string into substrings by using regular expressions.

2. Lemmatization

The classical order of NLP preprocessing is the Tokenization first and then the lemmatization of text. Lemmatization is the process that consists to map a token. Many programming languages, including C, R, Python, Ruby, Java, and Clojure have libraries or plugins that can lemmatize English words. In our system, we use Python.

For example, the lemmatization of the sentence: “My email has been hacked” is [‘email’, ‘have’, ‘be’, ‘hack’]

3. Stopwords removal

Stopwords removal is a common technique used in NLP preprocessing. This technique consists of removing the frequently used words called stopwords in the natural language. These words are usually removed before NLP preprocessing tasks such as classification and Topic Modelling and Text Classification.

The examples of English stop words are “I”, “you”, "the", "is", "my", "on", …, etc.

These words can be removed without usually changing the value or the meaning of the useful information during NLP preprocessing. Removing stop words improves the analysis because the

[1] http://nlp.stanford.edu/software/tokenizer.shtml.
[2] https://opennlp.apache.org/documentation/manual/opennlp.html#tools.tokenizer.
[3] https://www.nltk.org/

program can focus on the most relevant words related to the main subject. The predefined stopwords provided by Python libraries such as NLTK and Scikit Learn[4] can be used during the preprocessing tasks. This system uses a customized stopword list to keep only useful information for the classification.

Here is an example of our customized stopword list:

```
stop_words = stopwords.words('english')

stop_words.extend (['involved', 'suspect', 'give', 'tips', 'use', 'tell', 'properly',

                'think','believe','exactly','means', 'ultra','methods','get','rid',

                'remove','make','mac','definition','thank','please','advance'])
```

4. Keyword extraction

Natural language processing (NLP) is an important task of automatically extracting and summarizing information from text data. Keyword extraction is an important task in natural language processing. This enables us to represent user input in a condensed way. In our system, the compact representation of user input is used as a feature for classification models. The system will build a category prediction model, enriched by a set of a list of keywords.

5. Category prediction model

Here, after the data preprocessing and keyword extraction, let's apply the category prediction model in order to find the right category related to the user input. Depending on the predicted category, the model will find the appropriate question to ask. Actually, the user input could refer to more than one category. However, due to the limited information in our dataset, multi-label classification is hard to achieve. We intend to improve this and predict more than one category.

After preprocessing the user input, the next step it's to calculate the weight of each word in the user query. Each word is tested to know if it belongs to the keywords present in the attack category. If so, the weight belonging to this category is increased using the weight of the input

[4] https://scikit-learn.org/

words. In the end, the model will choose the category with the highest weight, which means each user input belongs to a specific attack category with the highest weight.

Getting the user query, the proposed system needs to identify the category of the attack in order to understand the problem and help the user. For that reason, a category prediction model was developed. The prototype defines nine different categories of attacks: Data breach, Wireless attack, Social engineering, Network-based attack, Phishing, Ransomware, Man in the middle, DDoS, and Malware.

Algorithm 1 presents the pseudo-code of the category prediction.

Algorithm 1 CategoryPrediction

```
Input: UQ: User Query
       CAT: List of 9 elements for the categories of attacks in the dataset
       KW: List of keywords for each category
Output: category
Initialization:
1: List of weights for each category W = [0,0,0,0,0,0,0,0,0]
2: word: Query iterator
3: cat: Categories iterator
4: LUQ = Preprocessing (UQ)
5: TermFrequencies = TermFrequency (LUQ) // list of weights for each word in the user query
6: Begin
7:  for word in LUQ do                          // for each word in the user query
8:      for cat in CAT do
9:          if word in $KW_{cat}$ then          // if it's in a specific category list of keywords
10:             $W_{cat}$ = $W_{cat}$ + TermFrequencies (word)     // increase the category weight.
11: category = max(W)              // return the category that has the maximum weight
12: End
```

As described in the algorithm above, we consider the user input, the different categories of attacks, and the keywords for each category. Depending on the chosen category, the algorithm will continue to proceed with the rest of the algorithms.

It's important to accord some weight to each word in the user input in order to assess the value of each term to the query. To do so, we use the Term Frequency method, which measures how frequently a term occurs in a text. Learning those weights would be better. Even better train a non linear classifier like a transformer.

It is important to mention that learning these weights using deep neural networks architectures, will lead to more effective results. However, due to the size limitation of our labeled dataset as detailed in chapter 4 section 4.1. we cannot proceed with this type of learning method.

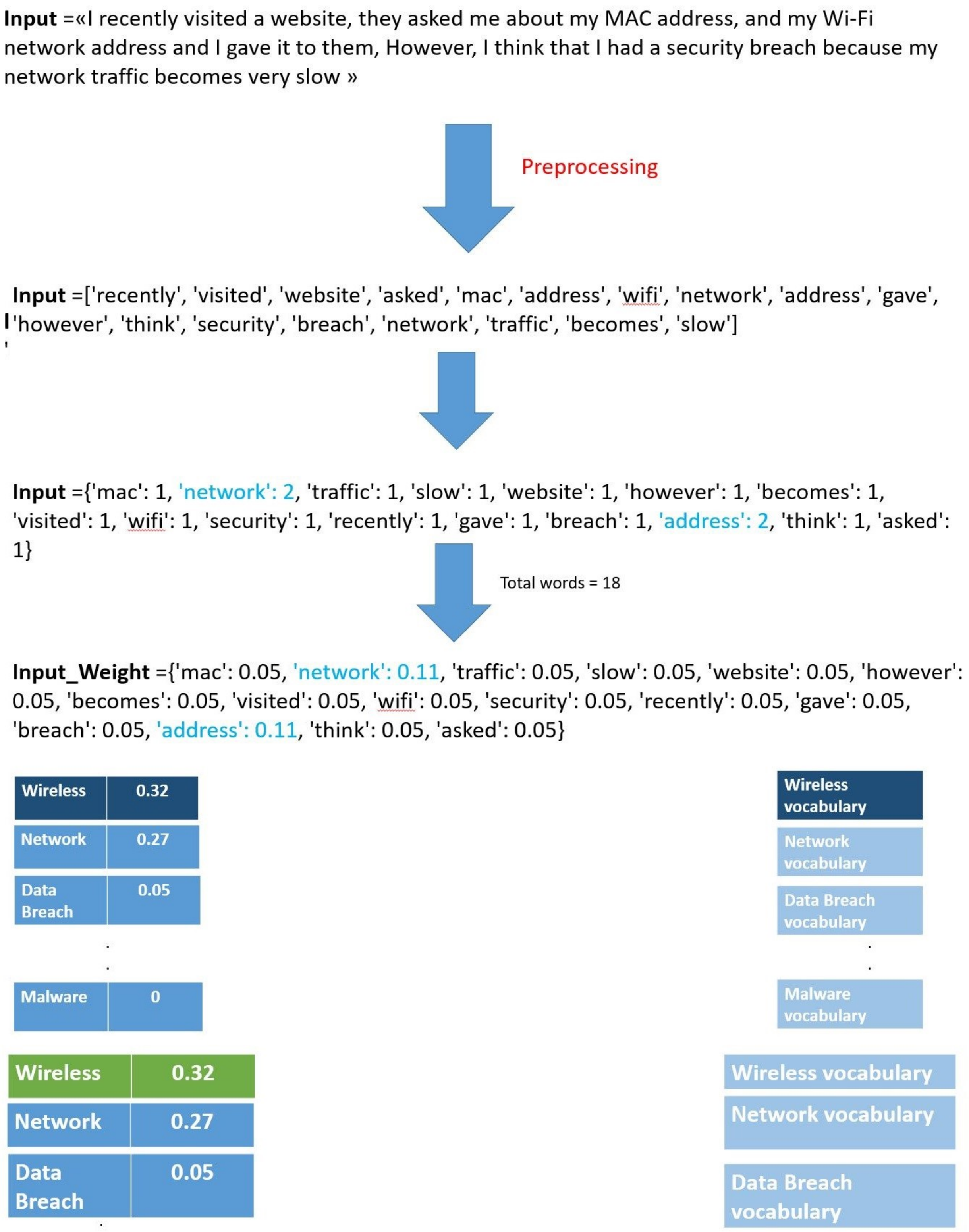
$$TF(t) = \frac{Number\ of\ times\ the\ term\ t\ appear\ in\ a\ text}{Total\ number\ of\ terms\ in\ the\ text} \tag{1}$$

To calculate the weight of a word in user input, we use the algorithm2: pseudo-code of the Term Frequency.

Algorithm 2 TermFrequency

```
Input: UQ: User Query which is a list of words.
Output: TermFrequencies
Initialization:
1: TermFrequencies = TermFrequency (UQ) // list of weights for each word in the user query
2: Total = numbers of words in UQ
3: Begin
4: for word in UQ do                                   // for each word in the user query
5:      TermFrequencies [word] = count (word, UQ) /Total
6: Return TermFrequencies
7: End
```

Illustration of the category prediction.



Input =«I recently visited a website, they asked me about my MAC address, and my Wi-Fi network address and I gave it to them, However, I think that I had a security breach because my network traffic becomes very slow »

Input =['recently', 'visited', 'website', 'asked', 'mac', 'address', 'wifi', 'network', 'address', 'gave', 'however', 'think', 'security', 'breach', 'network', 'traffic', 'becomes', 'slow']

Input ={'mac': 1, 'network': 2, 'traffic': 1, 'slow': 1, 'website': 1, 'however': 1, 'becomes': 1, 'visited': 1, 'wifi': 1, 'security': 1, 'recently': 1, 'gave': 1, 'breach': 1, 'address': 2, 'think': 1, 'asked': 1}

Input_Weight ={'mac': 0.05, 'network': 0.11, 'traffic': 0.05, 'slow': 0.05, 'website': 0.05, 'however': 0.05, 'becomes': 0.05, 'visited': 0.05, 'wifi': 0.05, 'security': 0.05, 'recently': 0.05, 'gave': 0.05, 'breach': 0.05, 'address': 0.11, 'think': 0.05, 'asked': 0.05}

Wireless	0.32
Network	0.27
Data Breach	0.05
.	
.	
Malware	0

Wireless	0.32
Network	0.27
Data Breach	0.05

Figure 25: Example of Category prediction

3.2.2 The user modeling

The user model can be considered as an abstract representation of the user in the system, so it contains all the information that helps the system to act according to the user. The information from the user model will be used later for matching the personalized recommendations.

In the proposed platform, the user model includes the user's ID, age, education level, work domain, and position. It also represents the current state of the user so that the system can know what problem he/she got before by consulting the storage in the user modeling module, as well as their feedback.

The first approaches of user modeling were implemented by Elain Rich in (1979) when the need for personalization has already been requested. The system also needs such user modeling for personalizing the solutions/recommendations.

In our case, a user model contains demographic information, like the age, education of a user, position, and work domain. The system exploits this model and, based on various techniques (collaborative-based filtering and content-based filtering algorithm), tries to find the match between users and recommendations/solutions. The user modeling step is the key task for personalized recommender systems, as the quality and success of our system depend on the correct representation of the user's model (*Gauch et al., 2007)*.

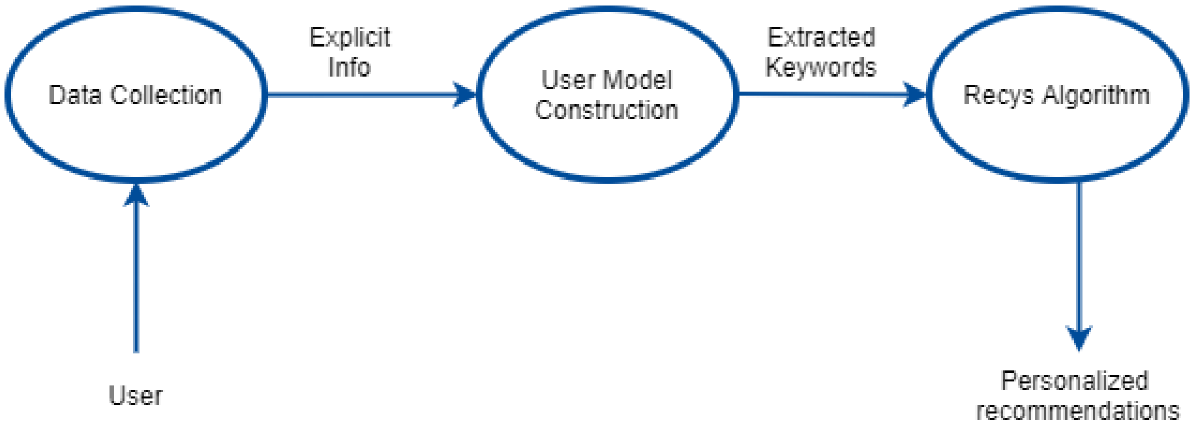
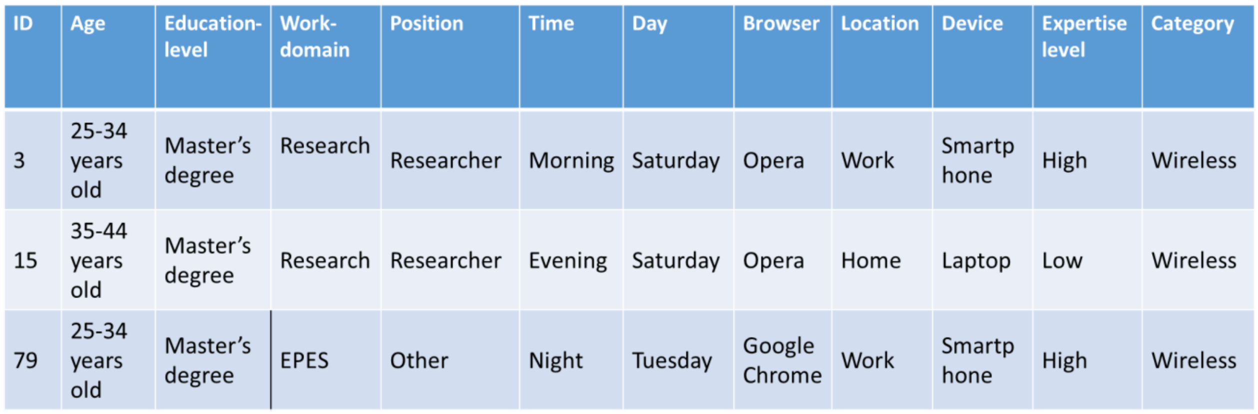


Figure 26: User modeling-based-personalized phase

We see the three main phases of building and exploiting the user model. First of all, we gather information about the user in the data collection phase. The data collection is done explicitly, which means the user provides to the system some personal information via a short informative question. This data is used for what they called initial profile and want to gather various information about the user's needs.

After the data collection phase is finished, the collected information has been transformed into an appropriate user model representation. The representation is done in form of weighted keywords. The generated user model is static, which means that the data is not going to be updated after the initial construction. The user model representation is responsible for storing the user's current state in an appropriate way. The user model represented as a vector contains some features, which are usually keywords from a text.

Once the user login, he/she provides some personal information such as Age, Education Level, Work domain, and Occupation. He/she will also provide some information about his/her context such as the Location, Time, Device, Browser, etc. We consider intuitively that every user has different cybersecurity levels and different interest levels concerning the eight user model elements mentioned above.



ID	Age	Education-level	Work-domain	Position	Time	Day	Browser	Location	Device	Expertise level	Category
3	25-34 years old	Master's degree	Research	Researcher	Morning	Saturday	Opera	Work	Smartphone	High	Wireless
15	35-44 years old	Master's degree	Research	Researcher	Evening	Saturday	Opera	Home	Laptop	Low	Wireless
79	25-34 years old	Master's degree	EPES	Other	Night	Tuesday	Google Chrome	Work	Smartphone	High	Wireless

Figure 27: example of user model and context data

It is essential to consider who these users are likely to be and what their characteristics suggest about their behavior. It is also important to consider their relevant knowledge and previous experience. As soon as the final user model representation has been constructed, in the third and last phase, the Recys algorithm exploits the data to recommend appropriate solutions to the user.

Before recommending a solution to users, it’s important to make sure the system understands their request through the NLP module and not giving him/her something that he/she got already. That’s why the system will ask some questions related to the user input category in order to better understand the problem and recommend something relevant. The category and recommendations are created by keywords.

3.2.3 Question asking module

The Cyberhelper provides a personalized question answering module that allows the system to interact with the user. Usually, the process of selecting questions is done by one of the following methods: popularity measures, keyword matching, frequencies of accessing documents, etc.

In this module, the system interacts with the user in order to better understand his/her request. In fact, after user input has been preprocessed, and certain keywords have been extracted, then the categorization is made to determine to which category of attacks the user input belongs.

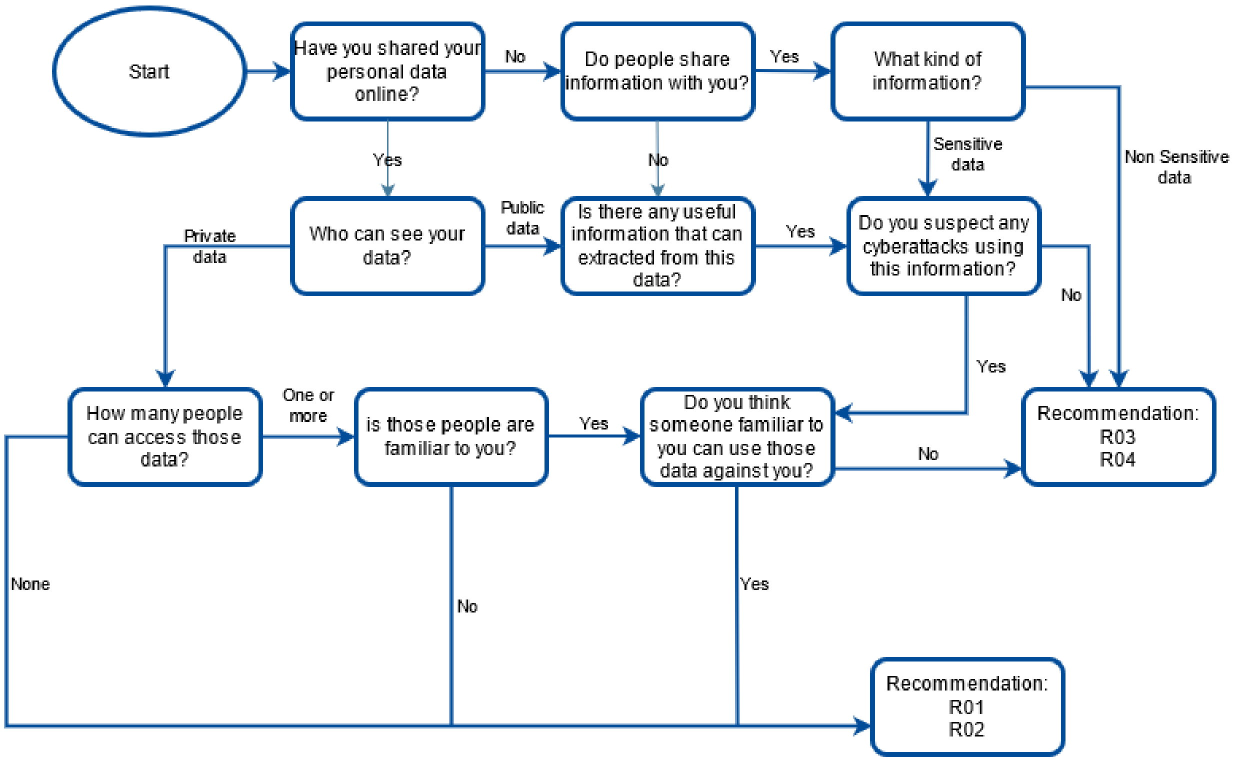

Start
Have you shared your personal data online?
No
Do people share information with you?
Yes
What kind of information?
Yes
No
Sensitive data
Non Sensitive data
Private data
Who can see your data?
Public data
Is there any useful information that can extracted from this data?
Yes
Do you suspect any cyberattacks using this information?
No
Yes
How many people can access those data?
One or more
is those people are familiar to you?
Yes
Do you think someone familiar to you can use those data against you?
No
Recommendation: R03 R04
None
No
Yes
Recommendation: R01 R02

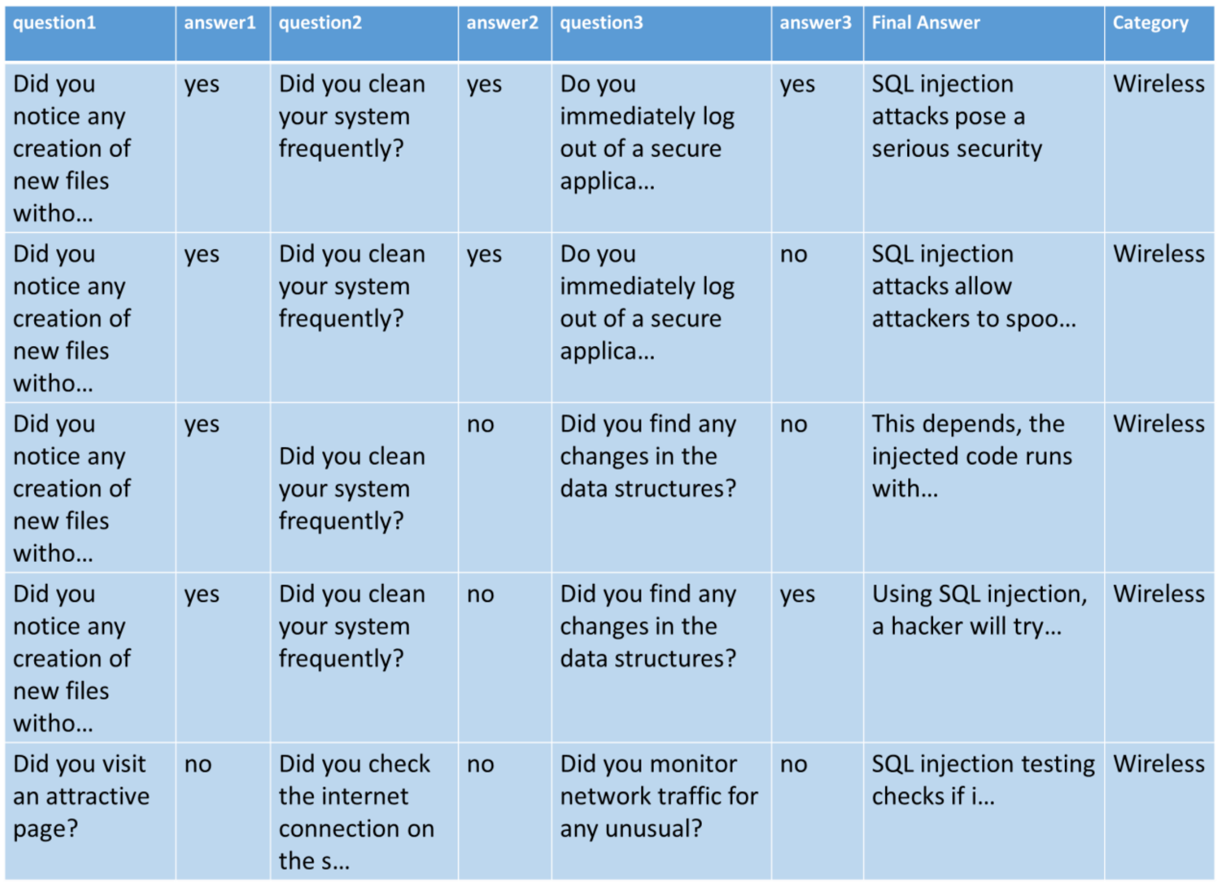
Figure 28: Example of question-based scenario

The keyword extraction can be considered as a bootstrap phase as the users do not have histories yet. Therefore, the keyword extraction process is based only on the user query. The user query reflects the topic-specific knowledge of the users but can be changed when a user asks a different question to the system. Inspired by the article (*Georgiadou et al., 2021*), cybersecurity considers the creation of a user model based on the 5 aspects of his/her life: (1) Age; (2) Education level; (3) Work domain; and (4) Position. The context contains contextual user information on 5 elements: (1) Location; (2) Device; (3) Time; (4) Browser, and (5) Day.

Over the years, the confirmation question has not been intensively studied, compared to other types of question answering. Truly speaking, it is not easy to get explicit evidence to conclude a proposition is false or true from an informative source. In our system, the confirmation questions or question-asking module require users' answers in the form of Yes or No. The system builds a tailor question asking module which first examines the user query, and then takes into consideration his/her user model and context in order to ask the most appropriate question related to the keyword extract from user queries. It classifies the user query by category of attacks. In fact, the classification performed in the user query affects the question asking module because a bad classification can generate a wrong set of system questions. The errors due to the miss-classification of user queries can also reduce the system's accuracy.

During the interaction between the system and the user, the user will only be able to answer questions asked by the system with a "yes", a "no" or a "not sure" depending on their knowledge. The system asking confirmation question follow the mentioned process: After extracting the most important keywords from the user's input, and applying the category prediction, the Cyberhelper asks the user a set of 3 questions, in order to better understand the target element. We decide to choose 3 set questions depending on the type of questions. For example, if the user asks a cybersecurity definition, Cyberhelper will only ask one question and provide a final answer. If the user asks a specific question about the issue that he/she is facing we will ask a set of questions to better understand his/her query.

Inspired by (Hamoud & Aimeur, 2020), the question-based system examines the user problem or issue through logical questions that users need to think about when they are using the internet. This question-based system is designed to help users to understand their actions online and consider the risks that these actions may bring. Users have to be aware when receiving an unknown email, downloading free software, or opening any link.



question1	answer1	question2	answer2	question3	answer3	Final Answer	Category
Did you notice any creation of new files witho...	yes	Did you clean your system frequently?	yes	Do you immediately log out of a secure applica...	yes	SQL injection attacks pose a serious security	Wireless
Did you notice any creation of new files witho...	yes	Did you clean your system frequently?	yes	Do you immediately log out of a secure applica...	no	SQL injection attacks allow attackers to spoo...	Wireless
Did you notice any creation of new files witho...	yes	Did you clean your system frequently?	no	Did you find any changes in the data structures?	no	This depends, the injected code runs with...	Wireless
Did you notice any creation of new files witho...	yes	Did you clean your system frequently?	no	Did you find any changes in the data structures?	yes	Using SQL injection, a hacker will try...	Wireless
Did you visit an attractive page?	no	Did you check the internet connection on the s...	no	Did you monitor network traffic for any unusual?	no	SQL injection testing checks if i...	Wireless

Figure 29: Example of question-based data

The Algorithm will start first by choosing a random question from system_question and then it will choose the next system questions (system question 2 and system question 3). Based on the value provided by the user for each question the algorithm will pick the more appropriate answer from the dataset. The user will rate the answer, if the rating is high then the algorithm ends, if not then the algorithm will pick another starting Question 'system question 1', and then do the whole process from the start.

This question-based system is a good opportunity to help users better understand and think about the risks of the internet. This represents an important part of the proposed system.

Therefore, it’s essential to make sure this will be designed coherently and logically to be relevant for the users.

3.2.4 The Recsys modeling

This section shows an algorithm that allows the system to perform the right recommendation, so each user who shares their cyber problems can get a personalized recommendation. Meanwhile, one of the critical aspects experienced by traditional question-based recommendation systems is the lack of mechanisms for selecting the appropriate recommendations. Usually, the process of recommendation is done based on the relationship and similarities between the users, or the users' historic preference data (*Zou et al., 2020*). To get the similarity between the users, the method used is to find the user's neighborhood and get predictions by neighbor’s rating (Bingqi, 2005). The second approach utilizes a user-preference-item tripartite graph model (Zhou & Han, 2019). In the two methods, the user can change or not provide his/her preferences.

To address the above-mentioned problem, the Cyberhelper uses a hybrid algorithm to provide optimal recommendation results according to its multiple algorithms.

a. Knowledge-based algorithm

This section explains the knowledge-based algorithm. Here, the system uses knowledge-based data that contains all the questions and answers for each category. Based on the predicted category, it will only choose the question related to this category, and depending on the user’s answer, it will get different scenarios. As mentioned throughout the introduction and the previous chapters, the Cyberhelper uses existing efficient algorithms to achieve its aim. Namely, a decision tree is at the core of algorithm 3 that is called the knowledge-based algorithm.

Algorithm 3 Knowledge-based algorithm

```
Input: category: Category of the attack,
       $QA$: Question and answers information
Output: $QA_{Rec}$: Recommendation of the algorithm
Initialization:
 1: Questions and Answers for a specific category, $QA_{category} = Ø$
 2: Questions and answers iterator: i = 0
 3: $QA_{Rec} = Ø$
 4: User answer, $a_i = Ø$
 5: $Q_i = Ø$,                          //ieth question in the datset
 6: $Q_{subset} = Ø$          //subset of QA given a question $Q_i$ and answers $a_i$
 7: Number of questions: N = 3 //In the knowledge database, there are three
    questions per scenario
 8: Begin
 9: $QA_{category}$ = select (QA, category) //get only data with the same category
10: $Q_{subset}$ = SelectRandom ($QA_{category}$)
11: $QA_{\{Q_{subset}\}}$                   //random scenario of questions
12: while $i <= N$ do //while the model didn't ask all the questions
13:     $Q_i = Q^i_{subset}$        // get the ieth question in the subset of knowledge
14:     $a_i$ = user response        //response of the user for the question $Q_i$
15:     $Q_{subset} = Q_{subset\{Q_i, a_i\}}$
16:     i = i+1
17: $QA_{Rec} = Q_{subset\{a_i\}}$
18: End
```

In other words, initially, the system has the knowledge database, which contains questions and answers for each category of attack, and for each question and answer scenario, there are multiple possibilities of recommendation. Knowing the category, as discussed above, Cyberhelper will filter the knowledge database, keeping only the questions and the answers database corresponding to that category. Given the filtered dataset, the algorithm starts by selecting a random question. After that, it will focus only on the rows that correspond to the selected question. Then, the system demands the user's answer and it will filter the new database based on the given answer. The Cyberhelper will ask the second question and gets the user's response. Doing the same process, it keeps only the rows that correspond to the user's answer. Finally, the system will ask the third question, and the final recommended answer will be the one that corresponds to the sequence of questions and answers. All of this is executed in a decision tree-like mechanism.

Illustration of the knowledge-based algorithm:

Given the user query, assuming that the predicted category is the wireless attack, we keep only the questions and answers of the wireless category as seen in Table 8.

Table 8: Example of knowledge-based data

question1	answer1	question2	answer2	question3	answer3	Final Answer	Category
Do you have any corroborating evidence?	yes	Recently, did you download any third-party sof…	yes	Did you notice any modifications in your system?	yes	You should ask your IT provider or someone…	Data Breach
Did you notice any creation of new files witho…	yes	Did you clean your system frequently?	yes	Do you immediately log out of a secure applica…	yes	SQL injection attacks pose a serious security	Wireless
Did you notice any creation of new files witho…	yes	Did you clean your system frequently?	yes	Do you immediately log out of a secure applica…	no	SQL injection attacks allow attackers to spoo…	Wireless
Did you notice any creation of new files witho…	yes	Did you clean your system frequently?	no	Did you find any changes in the data structures?	no	This depends, the injected code runs with	Wireless
Did you notice any creation of new files witho…	yes	Did you clean your system frequently?	no	Did you find any changes in the data structures?	yes	Using SQL injection, a hacker will try	Wireless
Did you visit an attractive page?	no	Did you check the internet connection on the s…	no	Did you monitor network traffic for any unusual…	no	SQL injection testing checks if i…	Wireless

The system selects a random question from the category predicted. Assuming that the first selected question is “Did you notice the creation of a new file without…”, and the user answer is “yes”. Then, we keep only the rows that correspond to the first question and the yes answer. The Cyberhelper will ask the second question and assume that the user will respond with ‘no’. We still keep only the rows that have the ‘no’ answer. Same, the system will ask the third question, and the user will respond with 'no'. An example of this is in Table 9.

Table 9: system asking question scenario

question1	answer1	question2	answer2	question3	answer3	Final Answer	Category
Did you notice any creation of new files witho...	yes	Did you clean your system frequently?	no	Did you find any changes in the data structures?	no	This depends, the injected code runs with...	Wireless
Did you notice any creation of new files witho...	yes	Did you clean your system frequently?	no	Did you find any changes in the data structures?	yes	Using SQL injection, a hacker will try	Wireless

The final answer will be the one that corresponds to the sequence of questions and answers. Table 10 shows an example of a final recommendation based on all the previous steps.

Table 10: Final recommendation from the Knowledge-based algorithm

question1	answer1	question2	answer2	question3	answer3	Final Answer	Category
Did you notice any creation of new files witho...	yes	Did you clean your system frequently?	no	Did you find any changes in the data structures?	no	This depends, the injected code runs with...	Wireless

After providing the recommendation to the user, he/she can reject or accept this recommendation. If the user rejects the recommendation, the system will give him/her a new one that needs to be different from the previous one. In order to ensure that the recommendation does not repeat in a session, The system will delete from the global question and answer dataset each question and answer that had been previously provided to this user. In a way that when it chooses a new random question, we will be sure that the algorithm did not choose a previous one.

b. Collaborative-based filtering algorithm

This section presents the collaborative-based filtering algorithm in order to get the top N similar user recommendation. Its main objective here is to return the recommended answers from the historical data of similar users who experience the same category of attack.

Algorithm 4 presents the pseudo-code of collaborative-based filtering using the nearest neighbour algorithm.

Algorithm 4 Collaborative-based filtering algorithm

```
Input:  AUI: All Users Information
        CAT: Category of the Attack
        CUI: Current user information
         N: Number of answers to keep
Output:  SUA:  Similar Users Answers
Initialization:
  1: Similarities = Ø: list of CUI similarities with different users
  2:  UC = Ø: List of users with the CAT attack category
  3: S = Ø:  Similarity between two users
  4: user: users list Iterator
  5: Begin
  6: UC = filter (AUI, CAT) // users who have similar attacks as the current user
  7: for  user in UC  do
  8:      S = cosineSimilarity (CUI, CAUI)
  9:      add S to similarities
 10:  SUA= Top (similarities, N)   // get the top N similar users recommendation
 11: End
```

To calculate the similarity between the current user and the other users, the system uses the cosine similarity method. Each user attribute is represented by a vector. The mathematical equation of Cosine similarity between two users A and B represented by two non-zero vectors, respectively, is:

$$Similarity = \frac{A.B}{\|A\|.\|B\|} \tag{2}$$

First, the system encodes both the current user and the other user's information. After that, it computes the cosine similarity score for the current user with each other user in the database, to finally get a list of top similar users.

In fact, the user attributes, as well as the context information, are categorical data that contain labels. In consequence, the cosine similarity can not operate on label data directly. It requires all input variables and output variables to be numeric. Therefore, the system applies the ordinal encoding technique in order to convert categorical data to a numerical form.

For example, the attribute “Age” can take the following values: 18-24 years old, 25-34 years old, 35-44 years old, 45-54 years old, 55-64 years old, and 75 years or older. After the encoding, these values will be 0,1,2,3,4,5, and 6 respectively. After that, we proceed with one-hot encoding, as shown in the examples below.

We illustrate the collaborative-based filtering algorithm.

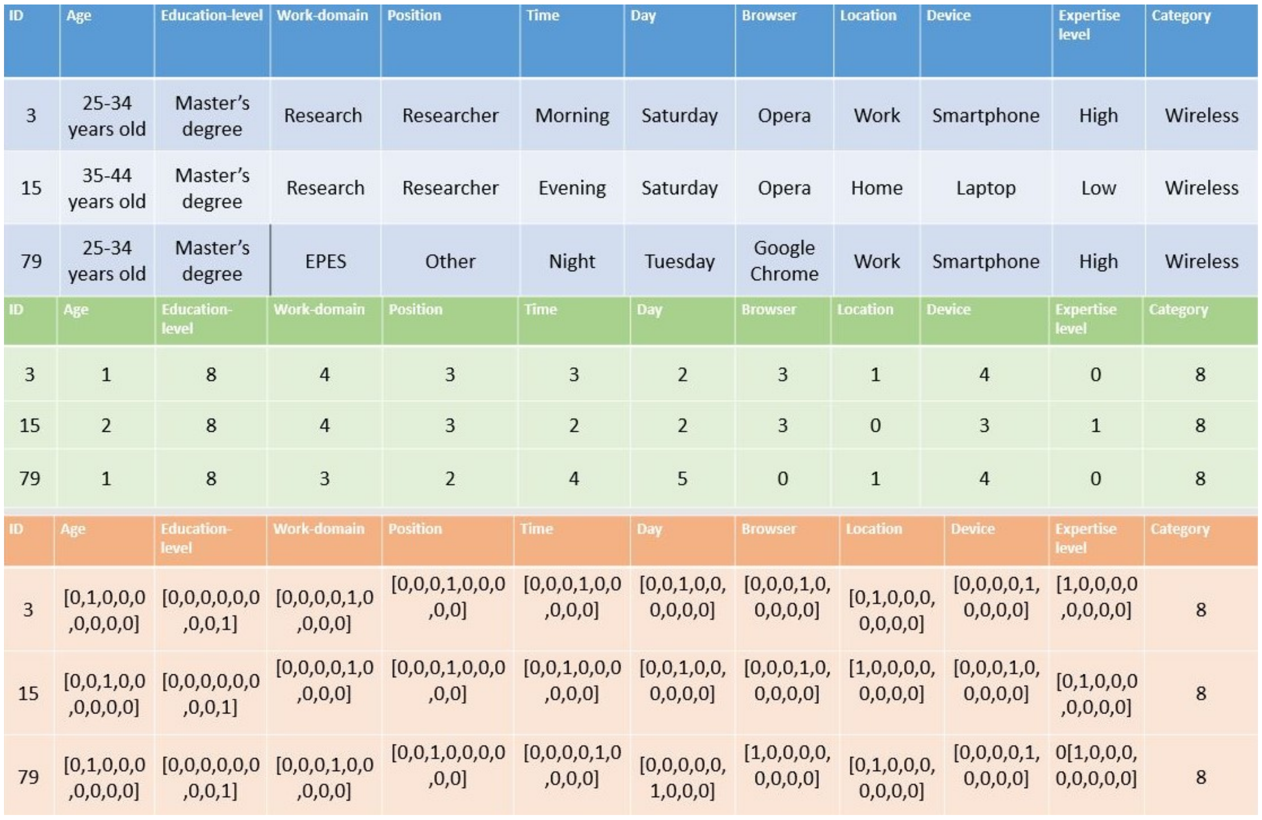
Consider the userID 2 with his/her information such as (age, id, Education level…) and his/her context information (time, day, device, …): an example of this is in Table 11. Before computing the similarity between this user and the other users, each feature needs to be encoded using the one-hot encoding technique.

Table 11: UserID 2 feature encoding

ID	Age	Education-level	Work-domain	Position	Time	Day	Browser	Location	Device	Expertise level	Category
2	25-34 years old	Master’s degree	Education	Researcher	Morning	Sunday	Opera	Home	Smartphone	High	Wireless
ID	**Age**	**Education-level**	**Work-domain**	**Position**	**Time**	**Day**	**Browser**	**Location**	**Device**	**Expertise level**	**Category**
2	1	8	4	3	3	3	3	0	4	0	8
ID	**Age**	**Education-level**	**Work-domain**	**Position**	**Time**	**Day**	**Browser**	**Location**	**Device**	**Expertise level**	**Category**
2	[0,1,0,0,0, 0,0,0,0]	[0,0,0,0,0,0, 0,0,1]	[0,0,0,0,1,0, 0,0,0]	[0,0,0,1,0,0, 0,0,0]	[0,0,0,1,0, 0,0,0,0]	[0,0,0,1,0, 0,0,0,0]	[0,0,0,1,0 ,0,0,0,0]	[1,0,0,0,0 ,0,0,0,0]	[0,0,0,0,1,0,0 ,0,0]	[1,0,0,0,0, 0,0,0,0]	[0,0,0,0,0, 0,0,0,1]

Also, we need to encode the features of all the other users using the same technique in order to be capable to compute the similarity. To simplify this step, Table 12 shows an example of user feature encoding.

Table 12: Users features encoding



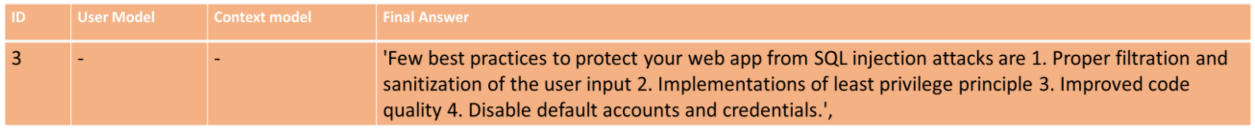
ID	Age	Education-level	Work-domain	Position	Time	Day	Browser	Location	Device	Expertise level	Category
3	25-34 years old	Master's degree	Research	Researcher	Morning	Saturday	Opera	Work	Smartphone	High	Wireless
15	35-44 years old	Master's degree	Research	Researcher	Evening	Saturday	Opera	Home	Laptop	Low	Wireless
79	25-34 years old	Master's degree	EPES	Other	Night	Tuesday	Google Chrome	Work	Smartphone	High	Wireless

ID	Age	Education-level	Work-domain	Position	Time	Day	Browser	Location	Device	Expertise level	Category
3	1	8	4	3	3	2	3	1	4	0	8
15	2	8	4	3	2	2	3	0	3	1	8
79	1	8	3	2	4	5	0	1	4	0	8

ID	Age	Education-level	Work-domain	Position	Time	Day	Browser	Location	Device	Expertise level	Category
3	[0,1,0,0,0,0,0,0,0]	[0,0,0,0,0,0,0,0,1]	[0,0,0,0,1,0,0,0,0]	[0,0,0,1,0,0,0,0,0]	[0,0,0,1,0,0,0,0,0]	[0,0,1,0,0,0,0,0,0]	[0,0,0,1,0,0,0,0,0]	[0,1,0,0,0,0,0,0,0]	[0,0,0,0,1,0,0,0,0]	[1,0,0,0,0,0,0,0,0]	8
15	[0,0,1,0,0,0,0,0,0]	[0,0,0,0,0,0,0,0,1]	[0,0,0,0,1,0,0,0,0]	[0,0,0,1,0,0,0,0,0]	[0,0,1,0,0,0,0,0,0]	[0,0,1,0,0,0,0,0,0]	[0,0,0,1,0,0,0,0,0]	[1,0,0,0,0,0,0,0,0]	[0,0,0,1,0,0,0,0,0]	[0,1,0,0,0,0,0,0,0]	8
79	[0,1,0,0,0,0,0,0,0]	[0,0,0,0,0,0,0,0,1]	[0,0,0,1,0,0,0,0,0]	[0,0,1,0,0,0,0,0,0]	[0,0,0,0,1,0,0,0,0]	[0,0,0,0,0,1,0,0,0]	[1,0,0,0,0,0,0,0,0]	[0,1,0,0,0,0,0,0,0]	[0,0,0,0,1,0,0,0,0]	0[1,0,0,0,0,0,0,0,0]	8

After calculation of similarity between the userID2 and the 3 users chose, it's clearly proved that the most similar user to userID2 is the userID3. So, the algorithm will select the final_answer from the userID3 similarly to how it is presented in Table 13.

Table 13: final recommendation result



ID	User Model	Context model	Final Answer
3	-	-	'Few best practices to protect your web app from SQL injection attacks are 1. Proper filtration and sanitization of the user input 2. Implementations of least privilege principle 3. Improved code quality 4. Disable default accounts and credentials.',

This concludes this section and next, the content-based filtering algorithm is detailed.

c. Content-based filtering algorithm

To provide personalization to our system, the authors (Zanker, 2019) are considered as an outline to guide us on how to build our system and create personalized recommendations. Personalization's main goal is to provide a tailored, relevant user experience to affect the level of user satisfaction. The design shape of a personalized system consists of four main dimensions described as follows: (1) User interface (Vaizman, 2018); (2) Content; (3) User model (Rahdari, 2020), and (4) Interaction process. All those dimensions can influence negatively or positively the way the system is perceived and used.

The design of the system has been built, based on the above four dimensions and also the machine learning techniques. In fact, machine learning is the main focus on the data used and the techniques or approaches to training and testing the models.

Since our system is targeting people from a different age range, we take into consideration their different backgrounds and the issue or problem that they can face online. In fact, with good or no cybersecurity knowledge, everybody is vulnerable to cyber attacks in some way.

The content-based filtering algorithm presents the personalization part of our system. In fact, it is a learnable algorithm that will continue learning each user interaction using the incremental learning method. At this stage, it already has the user information, the context information, and some historical user's data presenting their information, the context, and the recommended answers for each one of them.

Therefore, it can train an incremental adaptive random forest machine learning algorithm that will understand the distribution of users and contexts information in a way that it will capable of predicting the right recommendation for each user, as mentioned in the algorithm below.

The training set represents 75% of all the data with *673* training rows. Whereas, the test dataset is composed of *232* unique rows that have never been seen by the learning algorithm. The data used to train this model is manually labeled data, where the label defines the recommendation that the system should output to the user's query, the features are the user information, the contextual information, and the category of the attack. The model is trained using these features as detailed in chapter 4 in section 4.1.

Algorithm 5 IncrementalForestModel

```
Input: AUI: All Users Information
       ACI: All Context Information
       ARecom: All the previous Recommendations
       ADR: Adaptive Random Forest
Output: Learner: The trained forest learning algorithm
Initialization:
 1: contextual user information training data, Xtrain = AUI + ACI
 2: recommendations target data, ytrain = ARecom
 3: predicted recommendations, PR = Ø
 4: score of the trained model recommendation, score = 0
 5: The training data iterator, $X_i$ = Ø
 6: Learner = Ø                                    //the learning algorithm
 7: Begin
 8:   Learner = ADR (Xtrain)    //train the learning algorithm with Xtrain data
 9:   PR = predict (Learner, Xtrain)   //predict is a predefined method in ADRclass
10: score (ytrain, ypred)              //similarity score between PR and ytrain
11: while score (ytrain, PR) < 99% do
12:     for $X_i$ in Xtrain do
13:         if PR ≠ ytrain then
14:             Learner = Update model ($X_i$)
15: End
```

For the training phase, it's important to make sure that the model understands perfectly the distribution of the training data, by repeating those steps many times until the accuracy score reaches more than 99%. In other words, for each training step, we will evaluate the learning algorithm and if the score is less than 99%, then it will retrain the model only on the misclassified samples, by updating the model to better predict the misclassified samples.

It is important to highlight that we tried different training accuracy thresholds (90, 95 and 99), and each time, we evaluated our model using the unseen test data. The highest generalization results were achieved using the 99% training accuracy thresholds. That's why we define 99% as our threshold for the training accuracy.

In the end, the learning algorithm creates a tree structure that defines the information structure within the dataset. So, for a specific scenario, i.e, user information, context information, and the recommended answer, the algorithm creates a line that leads to the corresponded recommendation.

It is important to highlight that throughout this research, we used the default hyperparameters provided by the adaptative random forest incremental model. We only increased the number of trees within the random forest, in order to investigate if increasing the complexity of the random forest model will lead to better generalization results. But for the next future work, we intend to investigate the increasing complexity by fine tuning other hyperparameters rather than modifying only the number of trees within the random forest model.

Since we are using an ensemble technique, we evaluate the performance of the whole forest, not the performance of each tree in the forest. At each recommendation step, we pick the recommendation that has been provided by most of the independent trees. More statistical information about the evaluation of our incremental random forest model is defined in Chapter 4, in sections 4.3 and 4.4.

Throughout this research, we only focus on the incremental adaptive random forest model, since it is known to provide effective results for labeled classification tasks (Rojas et al., 2020). We did not investigate the performance of other incremental machine learning algorithms. However, for the future work, we intend to compare the performance of adaptive incremental random forest models with other incremental models for our classification task.

Now that we have the trained algorithm, we use it in the content-based filtering algorithm to predict the best recommendation for the current user. In the following, the pseudo-code of the content-based filtering is presented.

Algorithm 6 presents the pseudo-code of the content-based filtering.

Algorithm 6: Content-based filtering algorithm

```
Input: CUI: Current User Information //Age, Education level, Work domain, Position
Output: LRecom: Learner Recommendations
Initialization:
  1: Learning_al = IncrementalForestModel () //use the already trainedlearner
  2: LRecom = Ø
  3: Begin
  4: LRecom = predict (Learning_al, CUI) //predict is a method in the learning algorithm class
  5: End
```

The content-based filtering algorithm uses the result of the incremental learning algorithm to predict the answer depending on the current user information and the context. Therefore, the learning algorithm is a personalized algorithm as it predicts different recommendations even for the same user depending on the context.

d. Hybrid-based algorithm

The hybrid-based algorithm is the combination of three algorithms: knowledge-based, collaborative-based filtering, and content-based filtering.

According to the scenarios, different types of input can be used. In fact, knowledge-based recommendation relies on interactions with the users in the context of a knowledge base. Collaborative-based filtering recommendation relies on ratings and content-based filtering recommendation relies on textual description and users.

Algorithm 7 presents the hybrid algorithm.

Algorithm 7 Hybrid Algorithm

```
Input: AUI: All Users Information,
       UQ: User Query which is a list of words.
       QAData: System knowledge data
Output: $Hybrid_{Ranswer}$        //final recommendation of the hybrid algorithm
Initialization:
 1: RateThreshold = 3.5          // here, we put a threshold as an example
 2: AnswerFound = False
 3: URate = Ø                    //User recommendation rate
 4: rec = Ø                      //Recommendation's list iterator
 5: category = Ø                 //attack category for the user query
 6: $Cfa_{Recommendation}$ = Ø  // Collaborative filtering algorithm recommendation
 7: CbfaRecommendation = Ø // Content-based algorithm recommendation
 8: $Kba_{Recommendation}$ = Ø     // Knowledge-based algorithm recommendation
 9: $Hybrid_{Recommendations}$ = Ø //list of the three recommendation algorithms results
10: Begin
11: category = categoryPrediction ()
12: while AnswerFound   True and QAData ≠ Ø and AUI ≠ Ø do
13:     $Kba_{Recommendation}$ = Knowledge-based algorithm ()
14:     $Cfa_{Recommendations}$ = Collaborative-based filtering algorithm ()
15:     $Cbfa_{Recommendation}$ = Content-based filtering algorithm ()
16:     $Cfa_{Top}$ =Top ($Cfa_{Recommendations}$)
17:     AUI = $AUI_{\{Cfa_{Top}\}}$
18:     $Hybrid_{Recommendations}$ = $Kba_{Recommendation}$ ∪ $Cfa_{Recommendation}$ ∪
$Cbfa_{Recommendation}$
19:     for rec ϵ $Hybrid_{Recommendations}$ do
20:         URate = get rate from user
21:         if URate ≥ RateThreshold then
22:             $Hybrid_{RAnswer}$ = rec
23:             AnswerFound = True
24:             if rec ≠ $Cbfa_{Recommendation}$ then
25:                 AUI = AUI ∪ rec              // Save the recommendation
26:                 update Content-based filtering algorithm () // When the user
    selects $Kba_{Recommendation}$ or $Cfa_{Recommendation}$, then the incremental algorithm is
    updated so that next time it will take into account the user specific choice and it will
    give more precise recommendation
27:     Return $Hybrid_{RAnswer}$
28: End
```

This algorithm presents the interactions between the system and the user. Given the user query, the system tries to figure out the attack category. After that, it will return the results of the three algorithms. Each algorithm will return one possible answer. After that, it will return, in each step, the results of the three algorithms. Following this, the Cyberhelper requires the user to give some rating for each answer. Moreover, if this answer is not generated by the content-based filtering algorithm, then we need to update it. In this case, the model should be updated in order to consider the appropriate recommendation for the given user and context information, for further use.

Our system relies on the three algorithms to enhance the recommendation process since each one of them depends on a different type of information. The Knowledge-based Algorithm depends on the questions and answers dataset, the Collaborative-based filtering Algorithm is based on the similarity between users and the Content-based Filtering Algorithm works with the user and the context information.

Depending on these three algorithms, we can say that the system tries to provide a good recommendation because, with the available dataset, we cannot proceed with more complex methods such as the system learning which algorithm to choose, depending on the current situation. However, for the future work, we intend to design a learning model that will learn to rank the recommendations of the three different algorithms to finally predict the most relevant one for the user.

Illustration:

Given the user model and the context information, the recys module will execute the three algorithms: Knowledge-based, Collaborative based and Content-based filtering. Each algorithm will produce a specific recommendation. The Hybrid recommendation system takes into consideration these three recommendations. For each recommendation, the user will give a rating, and based on this rating of the recommendations, the system will decide a specific action to take. Assuming that the three recommendations are as follows in Table 14.

Table 14: Recommendation results of each algorithm

Algorithm	HybridRecommendations
Knowledge-based algorithm	KbaRecommendation: This depends, the injected code runs with ….
Collaborative-based filtering algorithm	CfaRecommendation: Few best practices to protect your web app from …
Content-based filtering algorithm	CbfaRecommendation: SQL injection attacks pose a serious security …

If the rate of one recommendation is higher than the threshold, the final recommendation will be that most rated recommendation. Here, the user gives the recommendation knowledge-based system the highest rate such as the ones given in Table 15. Then the final recommendation will be the knowledge-based recommendation.

Table 15: User rating each recommendation

Algorithm	HybridRecommendations	Rating
Knowledge-based algorithm	KbaRecommendation: This depends, the injected code runs with…	4
Collaborative filtering algorithm	CfaRecommendation: Few best practices to protect your web app from…	1
Content-based filtering algorithm	CbfaRecommendation: SQL injection attacks pose a serious security…	2

Now, assuming that for the three recommendations the user didn't choose any recommendation, that means he/she gives the same rating to all the recommendations. An example of this is in Table 16.

Table 16: Each recommendation with the same rate

Algorithm	HybridRecommendations	Rating
Knowledge-based algorithm	KbaRecommendation: This depends, the injected code runs with	1
Collaborative filtering algorithm	CfaRecommendation: Few best practices to protect your web app fro.	1
Content-based filtering algorithm	CbfaRecommendation: SQL injection attacks pose a serious security	1

In this case, the three recommendation algorithms will restart again, delete the provided recommendations in order to not Recommend the same rejected recommendations.

3.2.5 Domain Knowledge

This domain knowledge is the basis of all the information related to the system database, which is divided into solution or recommendation, system question, and category (cybersecurity attacks). It can be considered as the storage or organizational structure, where the information is kept, and use what we want if needed. It contains all the information about cybersecurity, including questions, definitions, categories of attacks, and solutions. Our domain knowledge will contain:

The list of questions is the set of labeled questions with titles based on the keywords provided to the system. For one category there are several solutions. For example, the subject "Phishing attack" can be related to the following questions: "how to protect me against the phishing attack?", "How to prevent phishing attacks?"

We understand here that the solutions are mini-tutorials giving clear instructions to the user, in our case recommendations. These are the questions that will be asked to the user to properly create a user model and place them in a specific context. These questions will influence the results of the recommendation.

The Cyberhelper has a database consisting of the cybersecurity solutions to be recommended and the features of these solutions. The users provide some sort of information useful for the system. Combining the knowledge-based data information with the user context, the system builds a user model. According to the information existing in a target user model and context, the system recommends suitable cybersecurity solutions to the user. Using all the features of cybersecurity solutions and user’s information can make better-personalized recommendations.

The cybersecurity solution is defined by its important features which can be described by the category of the attacks, the keyword, the system questions, …etc.

3.2.6 Recommendation result module

Once the recommendation process is done, each user will be able to get a personalized recommendation based on their user model, context, and user query. The proposed system provides an environment that enables users to understand and solve their cybersecurity issues and allows them to interact by text with the system.

A hybrid recommendation system uses a combination of knowledge-based recommendation, collaborative-based recommendation, and content-based recommendation systems to achieve high performance by reducing the drawbacks of the traditional recommendation techniques.

After the recommendation results of three algorithms are generated, the next step is to determine the weights of the appropriate results. In this system, all the different recommendation results have different importance, and they play important roles in the decision-making.

Each recommendation has a specific schema given the user and context information (given the values of each feature in the database). The weight of one of the recommendation results is closely related to the final recommendation result. If the result of one of the recommendation algorithms has higher accuracy, then the algorithm should be more reliable and influence the final decision process.

3.3 Implementation

The implementation phase is the final step of moving the project from concept to reality, to give us the actual project result. More specifically, since the "Cyberhelper" is a dynamic machine learning system, we used the following technologies:

- Python[5] is one of the most popular programming languages which can be used for other types of programming and software development besides web development. That includes backend development, software development, data science, and writing system scripts.
- Some Libraries such NLTK, Spacy, Numpy, Pandas,

[5] https://www.python.org/

- Scikit-multiflow library is a free and open-source machine learning package for output/multi-label and stream data written in Python.in Python.
- Scikit-learn is a free machine learning tool for predictive data analysis
- Jupyter Notebook[6] is free open-source software that allows to edit and run notebook documents via a web browser. It can be executed on a local desktop requiring no internet access or can be installed on a remote server and accessed through the internet.

The above-mentioned technologies are designed to work together, in order to implement our system.

The process of implementing our Cyberhelper starts by collecting and organizing user information, context, and cybersecurity information. It’s essential to know who the users are and what they are experiencing online. It’s also important to understand the connection between each user query the category of attacks present in the database. An NLP process can extract this information, transform it into a form understandable by the system.

Next, comes the recommender system implementation. Using common machine learning Python's scikit-learn[7] library, we are able to use the cosine similarity algorithm to compute this user set.

To summarize and reiterate, this chapter presents the architecture, methodology, and the overall foundation of the Cyberhelper which is the “personalized question-based cybersecurity recommendation system”. The Cyberhelper aims to help the user to solve the cyberattack that he/she is experiencing. This process starts with a prediction of the attack category from the user query which is NLP based.

The Cyberhelper, then, relies on its recommendation algorithm to personalize its output to the user. The said algorithm is in fact a hybrid with multiple well-established and pillars of the field

[6] https://jupyter.org/

[7] https://scikit-learn.org

including the knowledge-based algorithm, collaborative-based filtering algorithm, and content-based algorithm. The next chapter presents the proof of concept of the Cyberhelper.

Chapter 4 – Experimental Evaluation

This section designs a proof-of-concept experiment. The main goal of our experiment is to study the quality of the personalized recommendation. Throughout the evaluation step, we investigate to which degree our recommendation system can provide effective results when it encounters some unseen data. To ensure a proper design of the evaluation system, it's important to focus on both offline and online evaluation paradigms. In order to well understand the performance of the various recommendation algorithms implemented, different evaluation metrics and aspects were used.

The Knowledge-based algorithm, Collaborative-based filtering algorithm, and Content-based filtering recommendation algorithms will be evaluated on the fly using an online evaluation perspective, due to the fact that all those algorithms need the users' interaction to decide whether the recommendation is convenient or not. Besides the online evaluation, the Content-based recommendation algorithm could also be evaluated from an offline perspective. This recommendation component is based on an incremental machine learning algorithm that learns the user behavior to better predict the most convenient recommendation. This makes the use of machine learning statistical evaluation metrics such as precision, recall, f1-score, and accuracy more feasible to evaluate the Content-based filtering algorithm in an offline way. Throughout this section, the description of the management of the evaluation part of the recommendation system in both offline and online ways will be done.

4.1. Dataset

The experimental data were collected from multiple cybersecurity websites such as Kaspersky[8], Norton[9], Canada anti-fraud[10], … etc. The database includes *674* cybersecurity solutions, *9* categories of attacks, over *100* keywords per attack, and *255* system questions. After the data preparation step, for the offline evaluation, the dataset was divided into training and test sets[11].

[8] https://www.kaspersky.com

[9] https://ca.norton.com/

[10] https://www.antifraudcentre-centreantifraude.ca

[11] https://github.com/suzy91-ca/Personalized-question-based-recommendation-system.git

We use the training set in order to train our model, so the model will learn the distribution of our data. We use the testing set in order to evaluate if our model is capable to predict the right recommendation solutions.

To create the testing set, we choose the same user model but with different and unique context information, which is different from the context information presents in the training set. This is done in order to not overlap the training dataset. In fact, the testing dataset contains *273* rows. Our training dataset contains *673* rows.

Table 17: User model and context Dataset

Attribute	Meaning	Example
UserID	User unique id	[1,2, 3, ….,678]
Age	Range age of user	[18-44 years old, 25-34 years old, 35-44 years old, 45-54 years old, 55-64 years old, 65-74 years old, 75 years or older]
Education level	The highest degree or level of the school of the user	[Bachelor's degree, Master’s degree, Doctorate’s degree, Associate’s degree, Professional degree, No degree, …]
Work domain	The business domain of the organization	[Research, Education, Financial, Banking, Health, …]
Position	work position	[Manager, IT professional, Other, Researcher, Security professional]
Location	Where the attack happens	[Home, Work]
Day	When the attack happens	[Monday, Tuesday, Wednesday, Thursday, Friday, Saturday and Sunday]
Time	What time the attack happens	[Morning, Afternoon, Night, Evening]
Browser	Which browser is used by the user	[Google Chrome, Opera, Mozilla Firefox, Microsoft Edge]
Device	Devices used by the user	[Computer, Laptop, Smartphone, Tablet]
Level expertise	Cybersecurity level of user	[Medium, Low, High]
Final answer	Final recommendations (cybersecurity solutions)	[Hackers are constantly developing new types of malware and scams …]
Category	Category of attack	[Malware, Phishing, DDoS, Data breach, Network-based attack, Ransomware, Social engineering, wireless-based attack, and man in the middle]

In the following Table 18, we have the knowledge-based dataset used by the knowledge-based algorithm.

Table 18: Knowledge-based dataset

Attribute	Meaning	Example
System_question1	The first question asked by the system to the user.	[Do you have any corroborating evidence? …]
User_answer1	First answer from the user	Yes or No
System_question2	The second question asked by the system to the user	[Did you download any third-party software/application, recently?...]
User_answer2	Second answer from the user	Yes or No
System_question3	The third question asked by the system to the user	[Did you notice any modifications in your system?]
User_answer3	Third answer from the user	Yes or No
Final_answer	Final recommendations (cybersecurity solutions)	[Hackers are constantly developing new types of malware and scams …]
Category	Category of attack	[Malware, Phishing, DDoS, Data breach, Network-based attack, Ransomware, Social engineering, wireless-based attack, and man in the middle]

Table 19 is used by the category prediction algorithm.

Table 19: Category-Keyword dataset

Attribute	Meaning	Example
Category	Category of attack	[Malware, Phishing, DDoS, Data breach, Network-based attack, Ransomware, Social engineering, wireless-based attack, and man in the middle]
Keyword	Keyword of attack	Malicious, damage, attack, fraud, virus, …, etc

Because we have three recommendation components which are the knowledge-based algorithm, collaborative-based filtering, and content-based filtering. We are using three types of datasets: The knowledge-based dataset will be used by the knowledge-based algorithm, the user model and context dataset will be used by the collaborative-based and content-based filtering and the category-keyword dataset will be used by the category prediction algorithm.

4.2. Measures

This section presents the different metrics that will be used for the evaluation of our system.

- Accuracy: Accuracy measures, define the number of correct predictions divided by the total number of predictions. The accuracy helps us to compute the ratio of the number of correct predictions to the total number of input samples. By that, we can compute to which degree our model is capable of predicting the exact recommendations.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions\ made} \tag{3}$$

- Precision: In a binary classification the precision metric is the number of correct positive results divided by the number of positive results predicted by the classifier.

$$Precision = \frac{TruePositive}{TruePositive+FalsePositive} \tag{4}$$

In our recommendation task, we have a multi-classification task, that’s why we will not use the classical recall metrics, however, we will use the weighted recall metric:

$$WeightedPrecision = \frac{1}{Q}\sum_{j=1}^{Q}\frac{TruePositive}{TruePositive+FalsePositive} \tag{5}$$

Where Q defines the recommendation. This metric calculates the sum of precision for each recommendation and finds their average weighted value. We use this metric in order to report to which degree our learning model is capable of providing convenient recommendations.

- Recall: In a binary classification It is the number of correct positive results divided by the number of all samples that should have been identified as positive.

$$Recall = \frac{TruePositive}{TruePositive+FalseNegative} \tag{6}$$

In our recommendation task, we have a multi-classification task, that’s why we will not use the classical precision metrics, however, we will use the weighted precision metric:

$$WeightedRecall = \frac{1}{Q}\sum_{j=1}^{Q}\frac{TruePositive}{TruePositive+FalseNegative} \tag{7}$$

Where Q defines the recommendation. This metric computes the sum of recall for each recommendation and finds their average weighted value. We use the weighted recall in order to report to which degree our learning model is robust in terms of not providing false results.

- F1-Score: F1 Score is the Harmonic Mean between precision and recall. It tries to find the balance between precision and recall. The greater the F1 Score, the better is the performance of our model.

$$F1 = 2 \times \frac{1}{\frac{1}{precision} + \frac{1}{recall}} \tag{8}$$

In our recommendation task, we will use a weighted F1 score metric in order to take into consideration the multi-class information that we have.

$$WeightedF1 = 2 \times \frac{1}{\frac{1}{WeightedPrecision} + \frac{1}{WeightedRecall}} \tag{9}$$

Using the weighted F1-Score metric helps to report how many instances our learning algorithm predicted correctly, and also shows how robust the learning algorithm is. For the first part 'how many instances our learning algorithm predicted correctly' it's kind of similar to the accuracy metric but this metric also checks the robustness of the model. We will use it in order to compare its results with the accurate results. If there is a huge difference then we will know that our model is not robust.

The main intuition behind using these different 4 evaluation metrics, is that we want to make sure that our model produces trustful recommendations. We used the Weighted recall metric to see to which degree the model is robust and does not predict false results. We used the Weighted precision metric to report to which degree the learning model is predicting the convenient results. We used the Weighted f1 score metric to investigate to which degree the learning algorithm is capable of producing at the same time robust and precise results. We need to make sure that our learning algorithm is providing good results for both Weighted precision and Weighted recall, which can also be reported by the weighted f1-score metric. So, if we had good results for the different valuation metrics then we will know for sure that our model is performing well using the different recommendations. Throughout this section, we will describe how we managed to evaluate our recommendation system in both offline and online ways.

4.3. Offline Evaluation methods

Offline evaluation methods are, by far, the most common methods used to evaluate recommender systems from both the research and practice perspectives. Knowledge-based and collaborative-based filtering recommendation algorithms use the rule-based technique and cosine similarity respectively. These two algorithms are actually not learning anything because they only apply some rules and mathematical formulas to provide convenient recommendations. Evaluating these two methods can only be done with the interaction of the user which will decide whether the provided recommendation is sufficient or not. These two algorithms can be only evaluated in an online way. Throughout the offline evaluation of our system, it focuses more on the content-based recommendation algorithm, which uses an incremental machine learning algorithm in order to learn the exact behavior of a specific user.

The application of different evaluation metrics is important to truly reflect the effectiveness of the system, as one criterion can often provide an incomplete picture of the true performance of the recommender system. The authors (Aggarwal, 2016) mention that accuracy is one of the main metrics to evaluate a recommendation system. This system also uses different secondary metrics such as Weighted precision, Weighted recall, and Weighted f1 score as a combination metric to be more confident about the results found. It is important to highlight that due to the fact that there is a multiclassification problem, we used an average weighted version of these metrics. It's also necessary to investigate the effect of increasing the incremental learning algorithm hyper-parameters as well as the training steps number.

We overfit the learning algorithm with the training data and report the learning algorithm performance on the unseen data. This overfitting step is inspired by Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal in the PNAS published paper entitled 'Reconciling modern machine learning practice and the bias-variance trade-off *(Belkin et al., 2019*) where they propose a theory to explain the improved the generalization of very large models. In this article, the authors try different machine learning and deep learning algorithms. One of the algorithms that analyze is the Random Forest algorithm. This work uses an incremental extension of the Random Forest algorithm in the content-based recommendation algorithm part. It may be useful to train larger forests and report their generalization results. During the evaluation step, the test dataset contains *232* unique rows that have never been seen by the learning algorithm. The test dataset represents *25%* of all the data when the training data represent 75% of all the data with *673* training rows.

The training set and testing set were not divided randomly. We tried to divide them randomly then train and test the model using this random division, but due to the small amount of data when we divided the data randomly we saw that the random division can produce a split where some recommendations will be present only on the testing data which mean that the model have no clue about the existence of that recommendation, so whenever we want to use the test set to evaluate the model we will encounter some unseen recommendations, which will result in poor effectiveness values. It's like training a model on predicting the gender of a person using a training set that contains only 'Male' and' Female', but you test it on a test set that contains male, 'Female', and 'Other'. Because the model was trained only on 'Female' and 'Male' categories, when it encounters 'Other' in the test set it will predict it as 'Male' or 'Female' which will lead to a poor accuracy result. By splitting the dataset randomly, we encountered the same problem. We found that some recommendation in the test set is not seen by the model due to the small amount of data that we have. For this reason, we tried to split our data in a more effective way by ensuring that the same type of recommendation that we will evaluate our model on, are presented in the training dataset this is known as stratified sampling where during the data splitting, the labels will be selected with the same proportion for both training set and testing set. Using the training and testing data set, we were able to train and evaluate our model in an effective way. Throughout the training phase, we designed four different training categories, where each category defines a specific scenario.

Table 20 and Table 21 explain in detail the different results obtained from the different evaluation scenarios.

- **First scenario:** Throughout this evaluation scenario, *10* different classification trees are used within our incremental Adaptive random forest learning algorithm. Each classification tree is trained on a different data subset using the bootstrap technique. We train this model using the training set to finally evaluate it on the test set. It is important to highlight that throughout the evaluation of our model, we only used the test set. The results provided in each scenario could be biased results because the distribution of the test set is very similar to the distribution of the training data. We only use the test set for evaluating our model due to the fact that we don't have enough data to split it into three different subsets (training, test, and validation sets). However, we made sure that the test set will not be seen during the training phase so that the testing results can be trustable. We tried to reduce the amount of training data and leave 20% for the validation set. However, reducing the training data leads to a decrease in the effectiveness results. Consequently, we relied only on the test set to evaluate the model and we are aiming to investigate the effectiveness of other data for future experiments when we have more data to use.

- **Second scenario:** For the second evaluation scenario uses the same properties as the first scenario. However here, the complexity of the learning model is updated by increasing the number of tree estimators from *10 classification* trees to *100* classification trees, to see whether increasing the complexity of the model will lead to a better generalization.
- **Third scenario:** Throughout this evaluation scenario uses the same properties as the first scenario. However here, *25%* of the size of the training data is increased for the learning algorithm from *673* rows to *904* training rows. This is done in order to see whether increasing the training data will lead to better generalization results.
- **Fourth scenario:** For this evaluation scenario, let use the same properties as the first scenario with different classification trees and the same training data. However here, the learning algorithm will be trained many steps until it overfits the training data with an accuracy of more than *0.994*. We overfit the model in order to see if it can reach the modern regime shows in the article (*Belkin et al., 2019*) where overfitting the model leads to better generalization results.

Table 20: Training evaluation results for each scenario

	Scenario properties	**Train accuracy**	**Train Weighted-recall**	**Train Weighted-precision**	**Train Weighted-f1score**
First scenario	Training data and 10 estimators	0.87964	0.87964	0.83927	0.8492
Second scenario	Training data and 100 estimators	0.99702	0.99702	0.99702	0.99653
Third scenario	More Training data and 10 estimators	0.96879	0.96879	0.96594	0.96285

Fourth scenario	Training data, 10 estimators, and overfitting the model	**0.99405**	**0.99405**	**0.99554**	**0.99356**

Table 20 presents the results of the training evaluation of each scenario and Table 21, the results of the testing evaluation for each scenario.

Table 21: Testing evaluation results for each scenario

	Scenario properties	**Test accuracy**	**Test Weighted-recall**	**Test Weighted-precision**	**Test Weighted-f1score**
First scenario	Training data and 10 estimators	0.65984	0.65984	0.64022	0.64583
Second scenario	Training data and 100 estimators	0.78879	0.78879	0.78318	0.78448
Third scenario	More Training data and 10 estimators	0.91379	0.91379	0.91810	0.91379
Fourth scenario	Training data, 10 estimators and overfitting the model	**0.89224**	**0.89224**	**0.88491**	**0.88711**

The evaluation results of Table 20 and Table 21 show that the learning algorithm is capable of providing effective results in most of the evaluation scenarios.

- The first scenario shows that the non-complex learning algorithm which had been trained only once with only *10* different classification trees is capable to provide *0.87* training accuracy and *0.65* testing accuracy, which highlights the fact that in this work a simple model is not capable to provide good generalization results.
- The second scenario shows that both training and testing metrics results have increased. This highlights the fact that increasing the complexity of the learning algorithm leads to better generalization results.
- The third scenario proves that increasing the data size will also lead to better generalization results. The accuracy of the test dataset increased from *0.78* to *0.91* by only increasing the training data size. This highlights the importance of the data amount uses during the training phase. The biggest the data is, the more important the accuracy increases.

- The fourth scenario shows the effect of overfitting the model by training it to reach more than *0.995* of accuracy on the training data. comparing the fourth scenario and the first one, which has the same training properties, the observation made is that overfitting the incremental model leads to better generalization results. By training the content-based filtering algorithm many steps in a way to overfit the training data, this overfit will lead to better generalization results which support the modern regime idea proof described within (*Belkin et al., 2019*) paper.

Figure 4 reports the effect of overfitting the incremental learning algorithm on the training and testing sets. Also, it shows in a visual way the different results of the fourth scenario. At the first three training steps, we observe a huge difference between the training and testing accuracy, which is due to the fact that we are only training the model to increase the training accuracy without putting any constraint for the testing accuracy that it also should increase. So the first three learning steps define the normal bias-variance trade-off for training classical regimes (Belkin et al., 2019). However, if we continue training the learning algorithm for more steps to increase further more the training accuracy, we observe that increasing the complexity of our learning algorithm to a certain step will also lead to the increase of the testing accuracy. This is what had been defined by the modern regime in the (Belkin et al., 2019).

It is very important to highlight that the good generalization results that we had for the overfitted model could be related to the fact that the test set and training set have similar distributions, it is true that they both have different rows, but these rows have a similar distribution. The fact that we overfit our learning model on the distribution of the training set could lead to good results on the test set due to the similarity of the distribution. Because we only have a small dataset both for training and testing steps, we decided to more investigate this behaviour for future research steps when we collect more data. For now, we can not trust the results that we got in the fourth scenario because they can simply be driven from the distribution similarity between the test and the train set. However, we are aiming to more investigate this behaviour in future research stages.

The metrics results of the first scenario are reported in a more interpreted way in Figure 30.

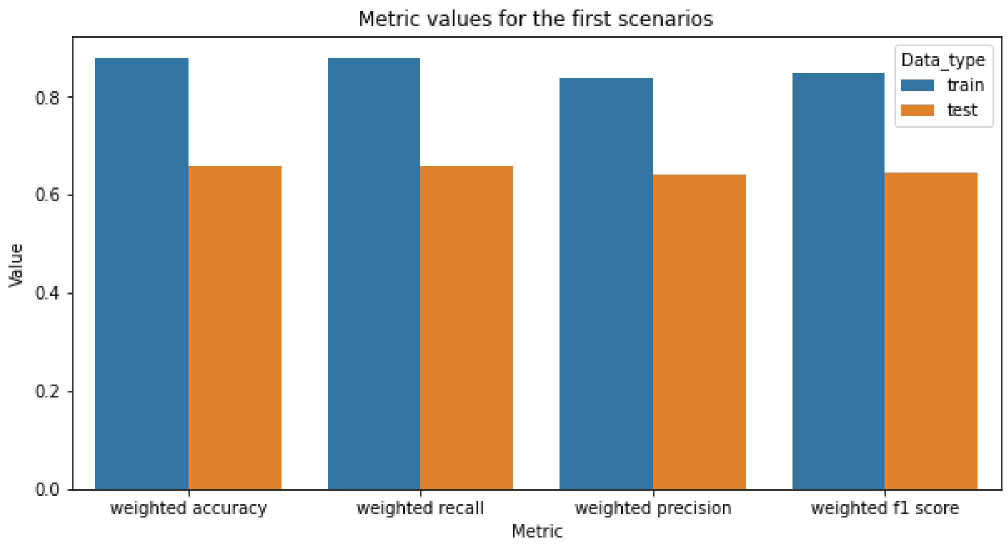


Figure 30: Training and testing metrics results for the first scenario

The results of the second scenario for each metric are reported in Figure 31.

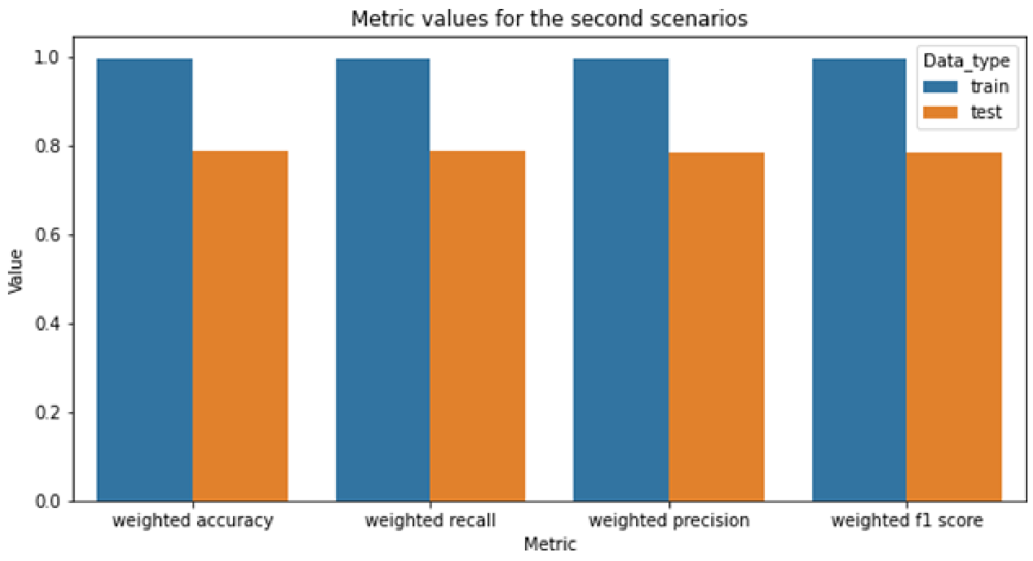


Figure 31: Training and testing metrics results for the second scenario

The results of the third scenario are reported in Figure 32.

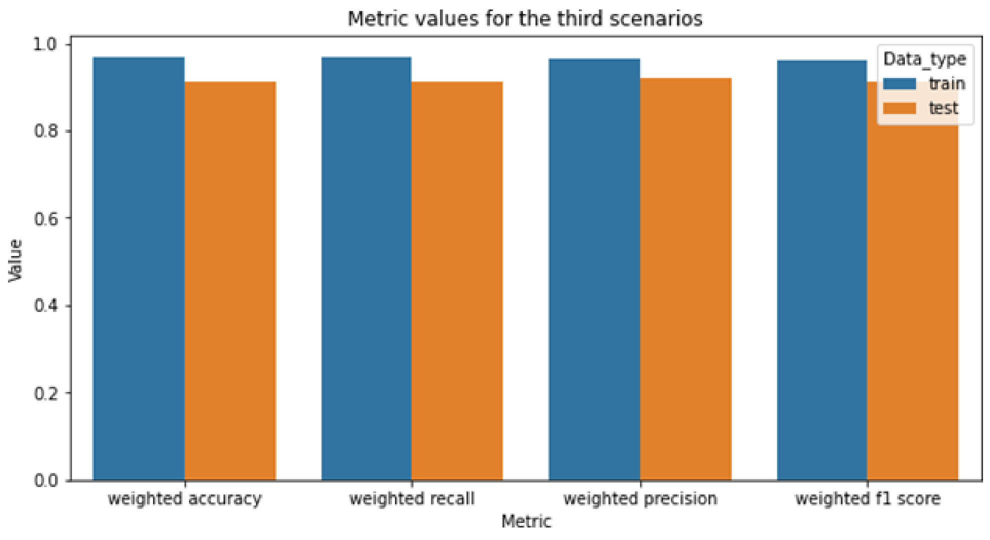


Figure 32: Training and testing metrics results for the third scenario

Figure 33 presents the effect of overfitting the incremental learning algorithm on the training and testing sets.

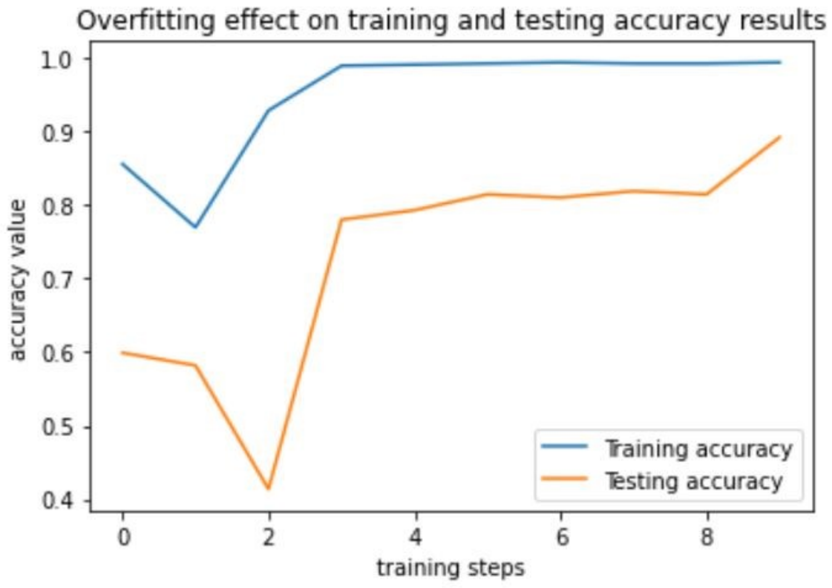


Figure 33: Training and testing accuracy for different training steps

By observing the evaluation table 20, table 21, and the graphical bar plots 1, 2, 3, and 4, we conclude that the best evaluation scenario is the third one with *0.96879* training accuracy and *0.91379* test accuracy. This is due to the fact that the third scenario model was trained with more training data amount comparing to the other scenarios. This highlights to which degree the amount of training data is important during the learning process. The more training data we use the more examples our learning model will learn and the more effective results we will have. Because the third scenario learned much more examples from the dataset, it was capable of providing more effective generalization results that's why we chose it to be the final model that we will use as the content-based filtering algorithm. It is important to highlight that scenario 1 and scenario 3 learning algorithms, have the same properties and the same hyperparameters. However, the third scenario was trained on more data.

Using Table 20 and Table 21, we observe that with additional training data for the same learning model, we were capable of increasing the test accuracy metrics from *0.65* to *0.91* which is a huge increase. This increase made us aware of the importance of the training data and more curious about the effect of using more data for training the two other scenarios, scenario 2 and scenario 4 in future research stages. Table 21 proves that using only a small amount of training data and increasing the complexity of the learning model, we achieve good generalization results. They are not as good as the results provided by more data amount, but they could be improved with more training data that's why we are curious to investigate the effect of training the complex models with more data. This investigation step will be achieved in future research stages.

4.4. Online Evaluation methods

The online evaluation paradigm defines a specific implementation of the evaluation metrics to measure user reactions with respect to the presented recommendations. Therefore, the participation of the user is essential in online systems. These testing paradigms are measuring the direct impact of the recommender system. However, it requires active user participation, so it could be used only during the deployment step. Throughout this part, it's important to explain in detail how to implement an online evaluation paradigm within our recommendation system to take into consideration user preference and interaction.

Due to the fact that this hybrid recommendation system was developed within a research environment, it is not evaluated yet with real users. Instead, an evaluation scenario was designed where we try to simulate a user behavior by giving the system some queries and observe the provided results. We then evaluate each algorithm manually by looking at the database to see whether the system is providing the appropriate recommendation or not. For the Knowledge-based algorithm, it shows that it’s better to focus on investigating the database manually to see whether the recommended solution is the convenient solution given the users' answers.

The Collaborative-based filtering algorithm focuses on investigating manually whether the system is returning the most similar users' recommendations.

For the online evaluation of the Content-based recommendation system, the algorithm is evaluated also depending on the preference of the users. If the user picks a recommendation from the three provided recommendations, then it will check manually if the picked recommendation is the same recommendation provided by the Content-based algorithm. If it's the case, then this means that this algorithm works as expected. However, if the picked recommendation is different from the recommendation provided by the Content-based algorithm, then this means that this algorithm is providing inconvenient results.   In this case, the algorithm will be updated on the fly to take into consideration the picked recommendation.

By simulating real users' behavior and by manually evaluating each algorithm, it found that each algorithm is giving the expected recommendation results. However, we are not sure of the real performance of these algorithms until exposing them to real users. So, the good performance of the recommendation algorithms will not be completely finished until deploying this system to real users, which is defined as a future goal.

Evaluating using both online and offline paradigms shows that our recommendation system is capable of providing an effective and personalized recommendation. Also using different offline evaluation metrics made us more confident about the results obtained in this work. However, besides the good results used in the test set, when evaluating our learning algorithm using a new dataset that contains a high level of randomness the learning algorithm was not capable of producing effective results. This is due to the complexity of the model and the size of the training data. As shown in Table 20 and Table 21 the more complex the learning algorithm is and the more training data, we use we will have more effective generalization results. So, as a perspective, For the future stages of this research, will focus on collecting more training data and increasing the complexity of the learning algorithm using the newly collected data to make the content-based recommendation algorithm capable of providing more precise recommendations.

Chapter 5 – Conclusion and Future Work

Recommendation systems are an important platform to provide personalized services. The traditional recommendation systems rarely use interactive methods to change the weights of the recommendation algorithms.

There are multiple ways to help to face cyber attacks. On the technological side, the technical experts, researchers, and government must consider some alternatives to prevent and protect users online. It’s also important to adopt a good approach to threat security awareness among the users who use diverse communication channels. There is a need for cybersecurity strategy and plans that are important for services and the application of multiple defense levels, and secure access provisioning.

The main objective of this research is to provide personalized question-based recommendation systems for cybersecurity preservation. More specifically, the following objectives are attained:

- Proposing cybersecurity domain knowledge that covers some aspects of cybersecurity such as the category of attacks, the keywords of each category, some definitions and cybersecurity solutions, and questions related to the cyberattacks. Section 3.1.5 further details this.
- Applying existing natural language processing tools and libraries to the user query in order to process the input and provide personalized output. This is needed for the Cyberhelper to “understand” the category of the issue the user is encountering.
- Using the existing mature and well-documented recommender system algorithms along with novel research on questing answering recommender systems to present the user with personalized responses to their quest. Section 3.1.4 explains all the different algorithms present in Cyberhelper.
- Providing proof of concept of both offline and online evaluation paradigms in order to better understand the performance of the various recommendation algorithms implemented.

In summary, this work presents personalized question-based cybersecurity recommendation systems. The Cyberhelper enables users to acquire the necessary knowledge on cybersecurity as well as prompt solutions in case they are currently in jeopardy.

To achieve these objectives, the Cyberhelper is founded on a hybrid algorithm combining Knowledge-based, Collaborative-based filtering, and Content-based filtering techniques The general process is divided into three steps, the establishment of a user model, find the nearest neighbors that are similar to the user, and generate recommendations based on the collective aforementioned information.

Although this study has potential for real-time deployment and paves the way for future research, the current personalized recommendation system needs improvement such as addressing the bias of self-reporting. Since the users present their current issue from their point of view and with their knowledge, mistakes can happen. A user might be panicking and asking the system why they cannot access their social media account and present information to the system assesses as they are being hacked. It might turn out to be something as simple as the user making a mistake entering their password and thinking someone has gotten access to it.

As future work, the proposed algorithm should be implemented with a larger database. In fact, it will be interesting to enrich the database with more information associated with the user model and context, increase the category of cyberattacks, develop a larger knowledge base data. In addition, the implementation of the system requires real users who will test the platform in real-time. In other words, the system will need more experiments with real users, provide acceptable response time, update the user model with every interaction, etc.

On the other hand, it’s important to be well aware that multiple other dimensions could be considered in implementing this system, such as the language of the users, culture, personality as well as the cyberattack situation (whether the cyberattack involves just one person or there are other factors), mandatory or voluntary, etc. Furthermore, the collaborative-based filtering algorithm mechanism can be improved by the system doing the ranking procedure automatically to accommodate large numbers of users.

In future research, creating a system that addresses different types of users within the same system will bring this work to the next stage. In order to achieve this, some solutions that include advanced machine learning approaches such as meta-learning or multi-view learning could be interesting perspectives to explore.

Also considering future work, we could investigate other machine learning techniques such as reinforcement learning to learn the optimal question-asking strategy. To improve the NLP implementation (in order to understand the user's input), more sophisticated methods might provide better results such as Bert for the classification task. The key advantage of such methods is their ability to achieve the best performance. Liu et al. (2017) report that the neural network can continuously extract useful features and filter out unusable ones.

Finally, consider as future work to add a chatbot as a "Cyberhelper" assistant to our system or in other words, work on a "conversational recommender system". It can be used to support and help the users who are facing any cyberattacks by providing an interactive user-friendly experience. The envisioned Cyberherper-bot could also evaluate or determine the risk or the level of the cyberattack and find related information from external sources to recommend it to the users through a recommendation engine mechanism.

Appendix

Training dataset

https://github.com/suzy91-ca/Personalized-question-based-recommendation-system/blob/main/Dataset/Training%20set.xlsx

Testing dataset

https://github.com/suzy91-ca/Personalized-question-based-recommendation-system/blob/main/Dataset/Test%20set.xlsx

Additional Training dataset

https://github.com/suzy91-ca/Personalized-question-based-recommendation-system/blob/main/Dataset/More%20data%20for%20training.xlsx

First scenario learning model notebook

https://github.com/suzy91-ca/Personalized-question-based-recommendation-system/blob/main/Notebook/Test10Estimator.ipynb

Second scenario learning model notebook

https://github.com/suzy91-ca/Personalized-question-based-recommendation-system/blob/main/Notebook/Test100_Estimator.ipynb

Third scenario learning model notebook

https://github.com/suzy91-ca/Personalized-question-based-recommendation-system/blob/main/Notebook/Test10_Estimator_MoreData.ipynb

Fourth scenario learning model notebook

https://github.com/suzy91-ca/Personalized-question-based-recommendation-system/blob/main/Notebook/Test10_Overfitted_Estimator.ipynb

Hybrid recommendation system notebook

https://github.com/suzy91-ca/Personalized-question-based-recommendation-system/blob/main/Notebook/HybridModelRecommendation.ipynb

Appendix: Example of Hybrid Algorithm

User Input:

Input =«I recently visited a website, they asked me about my MAC address, and my Wi-Fi network address and I gave it to them, However, I think that I had a security breach because my network traffic becomes very slow »

Step 1: Input preprocessing

Step 2: Outcome of recommendation: the results of the three algorithms can be as follow:

Algorithm	Recommendations
Knowledge-based algorithm	Rec 1: This depends, the injected code runs with
Collaborative filtering algorithm	Rec 2: Few best practices to protect your web appfro.
Content-based filtering algorithm	Rec 3: SQL injection attacks pose a serious security

The user chooses Rec 1. This ends the recommendation process.

Step 3: Satisfaction assessment

The user gets to rate the recommendation they received.

Algorithm	Hybrid Recommendations	Rating
Knowledge-based algorithm	Rec 1: This depends, the injected code runs with	4
Collaborative filtering algorithm	Rec 2: Few best practices to protect your web app fro.	1
Content-based filtering algorithm	Rec 3: SQL injection attacks pose a serious security	2

References

Aggarwal, C. C. (2016). *Recommender systems* (Vol. 1). Springer.

Aïmeur, E., Brassard, G., & Rioux, J. (2013). Data privacy: An end-user perspective. *International Journal of Computer networks and communications security*, *1*(6), 237-250.

Alabdulrahman, R. (2020). *Towards Personalized Recommendation Systems: Domain-Driven Machine Learning Techniques and Frameworks* Université d'Ottawa/University of Ottawa].

Alhijawi, B., & Kilani, Y. (2020). A collaborative filtering recommender system using genetic algorithm. *Information Processing & Management*, *57*(6), 102310.

Ammad-Ud-Din, M. a. I., Elena and Khan, Suleiman A and Oyomno, Were and Fu, Qiang and Tan, Kuan Eeik and Flanagan, Adrian. (2019). Federated collaborative filtering for privacy-preserving personalized recommendation system. *arXiv preprint arXiv:1901.09888*.

Andrews, E. (2015). Recommender systems for online dating.

Angiulli, F. a. F., Fabio. (2014). Exploiting domain knowledge to detect outliers. *Data mining and knowledge discovery*, 519-568.

Ayanouz, S., Abdelhakim, B. A., & Benhmed, M. (2020). A smart chatbot architecture based NLP and machine learning for health care assistance. Proceedings of the 3rd International Conference on Networking, Information Systems & Security,

Bai, X., Wang, M., Lee, I., Yang, Z., Kong, X., & Xia, F. (2019). Scientific paper recommendation: A survey. *IEEE access*, *7*, 9324-9339.

Ballal, S. K. (2018). Bumper to Bumper: Detecting and Mitigating DoS and DDoS Attacks on the Cloud, Part 2. Retrieved 31 May 2021, from https://securityintelligence.com/bumper-to-bumper-detecting-and-mitigating-dos-and-ddos-attacks-on-the-cloud-part-2/

Balog, K. a. R., Filip and Arakelyan, Shushan. (2019). Transparent, scrutable and explainable user models for personalized recommendation. Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval,

Beckers, K., & Pape, S. (2016). A serious game for eliciting social engineering security requirements. 2016 IEEE 24th International Requirements Engineering Conference (RE),

Belkin, M., Hsu, D., Ma, S., & Mandal, S. (2019). Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, *116*(32), 15849-15854.

Benavides, E. a. F., Walter and Sanchez, Sandra and Sanchez, Manuel. (2020). Classification of phishing attack solutions by employing deep learning techniques: A systematic literature review. *Developments and advances in defense and security*, 51-64.

Benhamdi, S. a. B., Abdesselam and Chiky, Raja. (2018). Personalized recommender system for e-Learning environment. *Education and Information Technologies*, *22*(4), 1455-1477.

Betancourt, Y. a. I., Sergio. (2020). Use of Text Mining Techniques for Recommender Systems. In *ICEIS (1)* (pp. 780-787).

Bingqi, Z. (2005). A collaborative filtering recommendation algorithm based on domain knowledge. *Computer Engineering*, *31*(21), 79-85.

Bokde, D., Girase, S., & Mukhopadhyay, D. (2015). Matrix factorization model in collaborative filtering algorithms: A survey. *Procedia Computer Science*, *49*, 136-146.

Brewer, R. (2016). Ransomware attacks: detection, prevention and cure. *Network Security*, *2016*(9), 5-9.

brooks, C. (2021). Alarming Cybersecurity Stats: What You Need To Know For 2021. *Forbes*. https://www.forbes.com/sites/chuckbrooks/2021/03/02/alarming-cybersecurity-stats-------what-you-need-to-know-for-2021/

Brundage, M., Avin, S., Clark, J., Toner, H., Eckersley, P., Garfinkel, B., Dafoe, A., Scharre, P., Zeitzoff, T., & Filar, B. (2018). The malicious use of artificial intelligence: Forecasting, prevention, and mitigation. *arXiv preprint arXiv:1802.07228*.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, *12*(4), 331-370.

Cai, W. a. C., Li. (2020). Predicting User Intents and Satisfaction with Dialogue-based Conversational Recommendations. Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization,

Canada.ca. (2021). *Canadian Anti-Fraud Centre*. https://antifraudcentre-centreantifraude.ca/

Corradini, I. (2020). Redefining the Approach to Cybersecurity. In *Building a Cybersecurity Culture in Organizations* (pp. 49-62). Springer.

Covington, P., Adams, J., & Sargin, E. (2016). Deep neural networks for youtube recommendations. Proceedings of the 10th ACM conference on recommender systems,

Cui, Z. a. X., Xianghua and Fei, XUE and Cai, Xingjuan and Cao, Yang and Zhang, Wensheng and Chen, Jinjun. (2020). Personalized Recommendation System Basedon Collaborative Filtering for IoT Scenarios. *IEEE Transactions on Services Computing*, *13*, 685-695.

Daniel Mikkelsen, H. S., and Malin Strandell-Jansson (2020). Privacy, security, and public health in a pandemic year. Retrieved 16 May 2021, from https://www.mckinsey.com/business-functions/risk/our-insights/privacy-security-and-public-health-in-a-pandemic-year

Davis, K. H., Biddulph, R., & Balashek, S. (1952). Automatic recognition of spoken digits. *The Journal of the Acoustical Society of America*, *24*(6), 637-642.

Domeniconi, G., Moro, G., Pagliarani, A., Pasini, K., & Pasolini, R. (2016). Job recommendation from semantic similarity of linkedin users' skills. International Conference on Pattern Recognition Applications and Methods,

Edelmann, D., Móri, T. F., & Székely, G. J. (2021). On relationships between the Pearson and the distance correlation coefficients. *Statistics & Probability Letters*, *169*, 108960.

Eisenberg, T. a. G., David and Hartmanis, Juris and Holcomb, Don and Lynn, M. Stuart and Santoro, Thomas. (1989). The Cornell commission: on Morris and the worm. *Communications of the ACM*, *32*(6), 706-709.

Emily Vogels, A. P., Lee Rainie and Monica Anderson. (2020). 53% of Americans Say the Internet Has Been Essential During the COVID-19 Outbreak. Retrieved 16 May 2021, from https://www.pewresearch.org/internet/2020/04/30/53-of-americans-say-the-internet-has-been-essential-during-the-covid-19-outbreak/

Fearnhead, P. a. R., Guillem. (2019). Changepoint detection in the presence of outliers. *Journal of the American Statistical Association*, *114*, 169-183.

Felfernig, A., Jeran, M., Ninaus, G., Reinfrank, F., Reiterer, S., & Stettinger, M. (2014). Basic approaches in recommendation systems. In *Recommendation Systems in Software Engineering* (pp. 15-37). Springer.

Ferrag, M. A. a. M., Leandros and Moschoyiannis, Sotiris and Janicke, Helge. (2020). Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. *Journal of Information Security and Applications*, *50*, 102419.

Ferreira, A., Coventry, L., & Lenzini, G. (2015). Principles of persuasion in social engineering and their use in phishing. International Conference on Human Aspects of Information Security, Privacy, and Trust,

Forouzandeh, S., Aghdam, A. R., Forouzandeh, S., & Xu, S. (2018). Addressing the cold-start problem using data mining techniques and improving recommender systems by cuckoo algorithm: a case study of Facebook. *Computing in Science & Engineering*, *22*(4), 62-73.

Gauch, S., Speretta, M., Chandramouli, A., & Micarelli, A. (2007). User profiles for personalized information access. *The adaptive web*, 54-89.

Georgiadou, A., Mouzakitis, S., & Askounis, D. (2021). Working from home during COVID-19 crisis: a cyber security culture assessment survey. *Security Journal*, 1-20.

Geuens, S., Coussement, K., & De Bock, K. W. (2018). A framework for configuring collaborative filtering-based recommendations derived from purchase data. *European Journal of Operational Research*, *265*(1), 208-218.

Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, *35*(12), 61-70.

Gomez-Uribe, C. A., & Hunt, N. (2015). The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)*, *6*(4), 1-19.

Guan, Z. a. J., Tongkai and Qian, Xu and Ma, Yan and Hong, Xuehai. (2018). A survey on big data pre-processing. 2017 5th Intl Conf on Applied Computing and Information Technology/4th Intl Conf on Computational Science/Intelligence and Applied Informatics/2nd Intl Conf on Big Data, Cloud Computing, Data Science (ACIT-CSII-BCD),

Guo, Y. a. W., Minxi and Li, Xin. (2017). An interactive personalized recommendation system using the hybrid algorithm model. *Symmetry*, *10*, 216.

Hadnagy, C., & Fincher, M. (2015). *Phishing dark waters*. Wiley Online Library.

Hamoud, A., & Aimeur, E. (2020). Handling User-oriented Cyber-attacks: STRIM, a User-based Security Training Model. *Frontiers in Computer Science*, *2*, 25.

Han, J. W., Hoe, O. J., Wing, J. S., & Brohi, S. N. (2017). A conceptual security approach with awareness strategy and implementation policy to eliminate ransomware. Proceedings of the 2017 International Conference on Computer Science and Artificial Intelligence,

Hannon, J., Bennett, M., & Smyth, B. (2010). Recommending twitter users to follow using content and collaborative filtering approaches. Proceedings of the fourth ACM conference on Recommender systems,

Hariharakrishnan, J. a. M., S and Kumar, KB Sundhara and others. (2017). Survey of pre-processing techniques for mining big data. 2017 International Conference on Computer, Communication and Signal Processing (ICCCSP),

He, L. a. J., Yan and Han, Weihong and Ding, Zhaoyun. (2014). Mining user interest in microblogs with a user-topic model. *China Communications*, *11*(8), 131-144.

Hickman, L. a. T., Stuti and Tay, Louis and Cao, Mengyang and Srinivasan, Padmini. (2020). Text preprocessing for text mining in organizational research: Review and recommendations. *Organizational Research Methods*, 1094428120971683.

Howe, M. (2009). Pandora's music recommender. *A Case Study, I*, 1-6.

Huang, Y.-F., & Wang, P.-L. (2017). Picture recommendation system built on Instagram. Proceedings of the 2017 International Conference on Artificial Intelligence, Automation and Control Technologies,

Hui, L. a. L., Haining and Zhang, Shu and Zhong, Zhaoman and Cheng, Jiang. (2019). Intelligent learning system based on personalized recommendation technology. *Neural Computing and Applications*, *31*(9), 4455-4462.

Hutchins, J. (1997). Fifty years of the computer and translation. *Machine Translation Review*, *6*(1997), 22-24.

Interpol. (2020). *COVID-19 cyberthreats*. Interpol. Retrieved 23 April 2021 from https://www.interpol.int/en/Crimes/Cybercrime/COVID-19-cyberthreats

Jain, A., Liu, I., Sarda, A., & Molino, P. (2019). Food Discovery with Uber Eats: Using Graph Learning to Power Recommendations. In: Uber Engineering Blog. https://eng. uber. com/uber-eats-graphlearning.

Jelodar, H., Wang, Y., Rabbani, M., & Ayobi, S. (2019). Natural Language Processing via LDA Topic Model in Recommendation Systems. *arXiv preprint arXiv:1909.09551*.

Jiang, P. R. Y., Zhan, H., & Zhuang, Q. (2010). Application research on personalized recommendation in distance education. 2010 International Conference on Computer Application and System Modeling (ICCASM 2010),

Johnson, J. (2021, Jannuary 25). *Development of malware worldwide 2015-2020*. Statista.

Johri, P., Khatri, S. K., Al-Taani, A. T., Sabharwal, M., Suvanov, S., & Kumar, A. (2020). Natural Language Processing: History, Evolution, Application, and Future Work. Proceedings of 3rd International Conference on Computing Informatics and Networks: ICCIN 2020,

Kaloudi, N. a. L., Jingyue. (2020). The ai-based cyber threat landscape: A survey. *ACM Computing Surveys (CSUR)*, *53*(1), 1-34.

Kanakaraddi, S. G., & Nandyal, S. S. (2018). Survey on parts of speech tagger techniques. 2018 International Conference on Current Trends towards Converging Technologies (ICCTCT),

Khan, N. A., Brohi, S. N., & Jhanjhi, N. (2020). UAV's applications, architecture, security issues and attack scenarios: a survey. In *Intelligent Computing and Innovation on Data Science* (pp. 753-760). Springer.

Kim, H.-N., Ha, I., Lee, K.-S., Jo, G.-S., & El-Saddik, A. (2011). Collaborative user modeling for enhanced content filtering in recommender systems. *Decision Support Systems*, *51*(4), 772-781.

Klavsnja-Milicevic, A. a. I., Mirjana and Vesin, Boban and Budimac, Zoran. (2018). Enhancing e-learning systems with personalized recommendation based on collaborative tagging techniques. *Applied Intelligence*, *48*(6), 1519-1535.

Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R., & Riedl, J. (1997). Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, *40*(3), 77-87.

Kumar, A. a. C., Jyotir Moy and Diaz, Vicente Garcia. (2020). A novel hybrid approach of SVM combined with NLP and probabilistic neural network for email phishing. *International Journal of Electrical and Computer Engineering*, *10*, 486.

Kumar, P., & Thakur, R. S. (2018). Recommendation system techniques and related issues: a survey. *International Journal of Information Technology*, *10*(4), 495-501.

Li, H. a. L., Haining and Zhang, Shu and Zhong, Zhaoman and Cheng, Jiang. (2019). Intelligent learning system based on personalized recommendation technology. *Neural Computing and Applications*, *31*, 4455-4462.

Li, Z., Fang, X., Bai, X., & Sheng, O. R. L. (2017). Utility-based link recommendation for online social networks. *Management Science*, *63*(6), 1938-1952.

Linden, G., Smith, B., & York, J. (2003). Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, *7*(1), 76-80.

Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., & Alsaadi, F. E. (2017). A survey of deep neural network architectures and their applications. *Neurocomputing*, *234*, 11-26.

Lu, J., Wu, D., Mao, M., Wang, W., & Zhang, G. (2015). Recommender system application developments: a survey. *Decision Support Systems*, *74*, 12-32.

Lu, X., & Du, Y. (2016). Interact-friend recommender system on Chinese Micro-blog based on structure balance. 2016 2nd IEEE International Conference on Computer and Communications (ICCC),

Luo, Y., Xiao, F., & Zhao, H. (2020). Hierarchical contextualized representation for named entity recognition. Proceedings of the AAAI Conference on Artificial Intelligence,

Madadipouya, K., & Chelliah, S. (2017). A literature review on recommender systems algorithms, techniques and evaluations. *BRAIN. Broad Research in Artificial Intelligence and Neuroscience*, *8*(2), 109-124.

Mao, J. a. B., Jingdong and Tian, Wenqian and Zhu, Shishi and Wei, Tao and Li, Aili and Liang, Zhenkai. (2018). Detecting phishing websites via aggregation analysis of page layouts. Procedia Computer Science,

Mieskes, M. (2017). A quantitative study of data in the NLP community. Proceedings of the First ACL Workshop on Ethics in Natural Language Processing,

Millecamp, M., Htun, N. N., Jin, Y., & Verbert, K. (2018). Controlling Spotify recommendations: effects of personal characteristics on music recommender user Interfaces. Proceedings of the 26th Conference on user modeling, adaptation and personalization,

Mohammed, H. J., Kasim, M. M., Hamadi, A. K., & Al-Dahneem, E. A. (2018). Evaluating of collaborative and competitive learning using MCDM technique. *Advanced Science Letters*, *24*(6), 4084-4088.

movielens.org. movielens.org. Retrieved 13 June from https://movielens.org/

Naumov, M. a. M., Dheevatsa and Shi, Hao-Jun Michael and Huang, Jianyu and Sundaraman, Narayanan and Park, Jongsoo and Wang, Xiaodong and Gupta, Udit and Wu, Carole-Jean and Azzolini, Alisson G and others. (2019). Deep Learning Recommendation Model for Personalization and Recommendation Systems. *arXiv preprint arXiv:1906.00091*.

Navigli, R. (2009). Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, *41*(2), 1-69.

Nilashi, M., Ibrahim, O., Yadegaridehkordi, E., Samad, S., Akbari, E., & Alizadeh, A. (2018). Travelers decision making using online review in social network sites: A case on TripAdvisor. *Journal of computational science*, *28*, 168-179.

oclc.org. *Kindred Works*. oclc.org. Retrieved 13 June from ttps://www.oclc.org/research/areas/data-science/kindredworks.html

Pan, R. a. Y., Tingsheng and Cao, Jianhua and Lu, Ke and Zhang, Zhanchao. (2015). Missing data imputation by K nearest neighbours based on grey relational structure and mutual information. *43*(3), 614-632.

Pan, Z., Cai, F., Ling, Y., & de Rijke, M. (2020). An Intent-guided Collaborative Machine for Session-based Recommendation. Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval,

Phishing.org. (2020). *Phishing Examples*. https://www.phishing.org/phishing-examples

phishing.org. (2021). *What Is Phishing?* https://www.phishing.org/what-is-phishing

PhishLabs. (2018). *2018 Phishing trends & intelligence report*. PhishLabs.

Pope, J. (2016). Ransomware: Minimizing the risks. *Innovations in clinical neuroscience*, *13*, 37.

Prakash, P. (2020). Extend Named Entity Recogniser (NER) to label new entities with spaCy. Retrieved 19 June 2020 from https://towardsdatascience.com/extend-named-entity-recogniser-ner-to-label-new-entities-with-spacy-339ee5979044

Prasad, K., & Kumari, M. (2020). A review on mathematical strength and analysis of Enigma. *arXiv preprint arXiv:2004.09982*.

Proofpoint. (2020). *2020 user risk report exploring vulnerability and behaviour in a people-centric threat*

landscape. Proofpoint.com. https://www.proofpoint.com/us/resources/white-papers/user-risk-report

Qin, C. a. Z., Hengshu and Zhu, Chen and Xu, Tong and Zhuang, Fuzhen and Ma, Chao and Zhang, Jingshuai and Xiong, Hui. (2019). Duerquiz: A personalized question recommender system for intelligent job interview. Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery \& Data Mining,

Rabaa Alabdulrahman, H. V., & Paquet, E. (2019). Active Learning and User Segmentation for the Cold-start Problem in Recommendation Systems.

Raghuraman, C. a. S., Sandhya and Shivshankar, Suraj and Chapaneri, Radhika. (2020). Static and dynamic malware analysis using machine learning. First International Conference on Sustainable Technologies for Computational Intelligence,

Rahdari, B. a. B., Peter and Babichenko, Dmitriy. (2020). Personalizing information exploration with an open user model. Proceedings of the 31st ACM Conference on Hypertext and Social Media,

Rai, A., & Borah, S. (2021). Study of various methods for tokenization. In *Applications of Internet of Things* (pp. 193-200). Springer.

Ravanelli, M., Parcollet, T., & Bengio, Y. (2019). The pytorch-kaldi speech recognition toolkit. ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),

Recommendation, A. U. P. t. O. F. i. M.-r. C. (2021). Adapting User Preference to Online Feedback in Multi-round Conversational Recommendation. Proceedings of the 14th ACM International Conference on Web Search and Data Mining,

Rekha, G. a. T., Amit Kumar and Reddy, V Krishna. (2018). A novel approach to solve class imbalance problem using noise filter method. International Conference on Intelligent Systems Design and Applications, Ribeiro-Navarrete, S. a. S., Jose Ramon and Palacios-Marqu{\'e}s, Daniel. (2021). Towards a new era of mass data collection: Assessing pandemic surveillance technologies to preserve user privacy. *Technological Forecasting and Social Change*, *167*, 120681.

Rich, E. (1979). User modeling via stereotypes. *Cognitive science*, *3*(4), 329-354.

Richardson, R., North, M. M., & Garofalo, D. (2021). Ransomware: The Landscape Is Shifting--A Concise Report. *International Management Review*, *17*(1), 5-86.

Rojas, J. S., Pekar, A., Rendón, Á., & Corrales, J. C. (2020). Smart User Consumption Profiling: Incremental Learning-Based OTT Service Degradation. *IEEE access*, *8*, 207426-207442.

Salema, R. B. a. A. i. m., Esma and Hageb, Hicham. (2020a). A Nudge-based Recommender System Towards Responsible Online Socializing. OHARS'20: Workshop on Online Misinformation- and Harm-Aware Recommender System,

Salema, R. B. a. A. i. m., Esma and Hageb, Hicham. (2020b). A Nudge-based Recommender System Towards Responsible Online Socializing. Proceedings of the Workshop on Online Misinformation- and Harm-Aware Recommender Systems,

Schafer, J. B., Konstan, J., & Riedl, J. (1999). Recommender systems in e-commerce. Proceedings of the 1st ACM conference on Electronic commerce,

Schedl, M. (2016). The lfm-1b dataset for music retrieval and recommendation. Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval,

Schwarz, D. (2019). ServHelper and FlawedGrace - New malware introduced by TA505 Retrieved 13 June 2021, from https://www.proofpoint.com/us/threat-insight/post/servhelper-and-flawedgrace-new-malware-introduced-ta505

Serrano-Guerrero, J., Herrera-Viedma, E., Olivas, J. A., Cerezo, A., & Romero, F. P. (2011). A google wave-based fuzzy recommender system to disseminate information in University Digital Libraries 2.0. *Information Sciences*, *181*(9), 1503-1516.

Sicari, S. a. R., Alessandra and Coen-Porisini, Alberto. (2020). 5G In the internet of things era: An overview on security and privacy challenges. *Computer Networks*, *179*, 107345.

Solangi, Y. A. a. S., Zulfiqar Ali and Aarain, Samreen and Abro, Amna and Mallah, Ghulam Ali and Shah, Asadullah. (2018). Review on Natural Language Processing (NLP) and its toolkits for opinion mining and sentiment analysis. 2018 IEEE 5th International Conference on Engineering Technologies and Applied Sciences (ICETAS),

Stein, S., & Jacobs, a. J. (2020, 16 March). Cyber-Attack Hits U.S. Health Agency Amid Covid-19 Outbreak. *Bloomberg*. https://www.bloomberg.com/news/articles/2020-03-16/u-s-health-agency-suffers-cyber-attack-during-covid-19-response

Su, D. a. L., Jiqiang and Wang, Xiaoyang and Wang, Wei. (2018). Detecting Android locker-ransomware on chinese social networks. *IEEE access*, *7*, 20381--20393.

Sun, S., Luo, C., & Chen, J. (2017). A review of natural language processing techniques for opinion mining systems. *Information fusion*, *36*, 10-25.

Sundermann, C. V., de Padua, R., Tonon, V. R., Marcacini, R. M., Domingues, M. A., & Rezende, S. O. (2020). A context-aware recommender method based on text and opinion mining. *Expert Systems*, *37*(6), e12618.

Symantec. (2020). Threat Landscape Trends – Q1 2020. *Symantec.com*. Retrieved June 9, from https://symantec-enterprise-blogs.security.com/blogs/threat-intelligence/threat-landscape-q1-2020

Tesfay, W. B. a. H., Peter and Nakamura, Toru and Kiyomoto, Shinsaku and Serna, Jetzabel. (2019). PrivacyGuide: towards an implementation of the EU GDPR on internet privacy policy evaluation. Proceedings of the Fourth ACM International Workshop on Security and Privacy Analytics,

TrendMicro. (2020). *Developing Story: COVID-19 Used in Malicious Campaigns*. Trend Micro. Retrieved 23 April from https://www.trendmicro.com/vinfo/us/security/news/cybercrime-and-digital-threats/coronavirus-used-in-spam-malware-file-names-and-malicious-domains

Tsai, C.-F. a. L., Miao-Ling and Lin, Wei-Chao. (2018). A class center based approach for missing value imputation. *Knowledge-Based Systems*, *151*, 124-135.

Vaizman, Y. a. E., Katherine and Lanckriet, Gert and Weibel, Nadir. (2018). Extrasensory app: Data collection in-the-wild with rich user interface to self-report behavior. Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems,

Wang, D., Liang, Y., Xu, D., Feng, X., & Guan, R. (2018). A content-based recommender system for computer science publications. *Knowledge-Based Systems*, *157*, 1-9.

Wang, D., Yih, Y., & Ventresca, M. (2020). Improving neighbor-based collaborative filtering by using a hybrid similarity measurement. *Expert Systems with Applications*, *160*, 113651.

Wang, Y., Wang, M., & Fujita, H. (2020). Word sense disambiguation: A comprehensive knowledge exploitation framework. *Knowledge-Based Systems*, *190*, 105030.

WHO. (2021). *Beware of criminals pretending to be WHO*. WHO. Retrieved 24 April from https://www.who.int/about/communications/cyber-security

Wu, L., & Grbovic, M. (2020). How Airbnb Tells You Will Enjoy Sunset Sailing in Barcelona? Recommendation in a Two-Sided Travel Marketplace. Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval,

Xu, K. a. Y., Jingxuan and Xu, Jun and Gao, Sheng and Guo, Jun and Wen, Ji-Rong. (2021). Adapting User Preference to Online Feedbackin Multi-round Conversational Recommendation. Proceedings of the 14th ACM International Conference on Web Search and Data Mining,

Yang, L. a. Z., Yu and Cai, Xiaoyan and Dai, Hang and Mu, Dejun and Guo, Lantian and Dai, Tao. (2018). A LSTM based model for personalized context-aware citation recommendation. *IEEE access*, *6*, 59618-59627.

Yang, N., Chen, L., & Yuan, Y. (2021). An Improved Collaborative Filtering Recommendation Algorithm Based on Retroactive Inhibition Theory. *Applied Sciences*, *11*(2), 843.

Yse, D. L. (2019). Your Guide to Natural Language Processing (NLP). Retrieved 06 June 2021, from https://towardsdatascience.com/your-guide-to-natural-language-processing-nlp-48ea2511f6e1

Yusri, R. a. A., Adel and A{\"\i}meur, Esma. (2020). Teens-Online: a Game Theory-Based Collaborative Platform for Privacy Education. *International Journal of Artificial Intelligence in Education*, 1-43.

Zanker, M. a. R., Laurens and Jannach, Dietmar. (2019). Measuring the impact of online personalisation: Past, present and future. *International Journal of Human-Computer Studies*, *131*, 160-168.

Zetter, K. (2017). What Is Ransomware? A Guide to the Global Cyberattack's Scary Method. Retrieved 31 May 2021, from https://www.wired.com/2017/05/hacker-lexicon-guide-ransomware-scary-hack-thats-rise/

Zhang, Q., Lu, J., & Jin, Y. (2021). Artificial intelligence in recommender systems. *Complex & Intelligent Systems*, *7*(1), 439-457.

Zhou, W., & Han, W. (2019). Personalized recommendation via user preference matching. *Information Processing & Management*, *56*(3), 955-968.

Zou, J., Chen, Y., & Kanoulas, E. (2020). Towards question-based recommender systems. Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval,