CAPITAL UNIVERSITY OF SCIENCE AND TECHNOLOGY, ISLAMABAD

On the Effectiveness of Private Information Retrieval Protocols

by

Rafiullah Khan

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the

Faculty of Computing
Department of Computer Science

2020

On the Effectiveness of Private Information Retrieval Protocols

By

Rafiullah Khan

(PC 133006)

DEPARTMENT OF COMPUTER SCIENCE
CAPITAL UNIVERSITY OF SCIENCE AND TECHNOLOGY
ISLAMABAD
2020

Dedicated to Loving Memory of my Father Dr. Shafiullah Khan and my Mother Mrs. Shafiullah Khan

CAPITAL UNIVERSITY OF SCIENCE & TECHNOLOGY ISLAMABAD

Expressway, Kahuta Road, Zone-V, Islamabad
Phone:+92-51-111-555-666 Fax: +92-51-4486705
Email: info@cust.edu.pk Website: https://www.cust.edu.pk

CERTIFICATE OF APPROVAL

This is to certify that the research work presented in the thesis, entitled "**On the Effectiveness of Private Information Retrieval Protocols**" was conducted under the supervision of **Dr. Muhammad Arshad Islam**. No part of this thesis has been submitted anywhere else for any other degree. This thesis is submitted to the **Department of Computer Science, Capital University of Science and Technology** in partial fulfillment of the requirements for the degree of Doctor in Philosophy in the field of **Computer Science.** The open defence of the thesis was conducted on **22 June, 2020.**

Student Name :	Mr. Rafiullah Khan (PC133006)	[illegible]

The Examination Committee unanimously agrees to award PhD degree in the mentioned field.

Examination Committee :

(a)	External Examiner 1:	Dr. Ayyaz Hussain, Associate Professor QAU, Islamabad	[illegible]
(b)	External Examiner 2:	Dr. Fawad Hussain, Associate Professor GIKI, Topi, Swabi	[illegible]
(c)	Internal Examiner :	Dr. Amir Qayyum Professor CUST, Islamabad	[illegible]

Supervisor Name :	Dr. Muhammad Arshad Islam Associate Professor FAST-NUCES, Islamabad	[illegible]
Name of HoD :	Dr. Nayyer Masood Professor CUST, Islamabad	[illegible]
Name of Dean :	Dr. Muhammad Abdul Qadir Professor CUST, Islamabad	[illegible]

AUTHOR'S DECLARATION

I, **Mr. Rafiullah Khan (Registration No. PC13006)**, hereby state that my PhD thesis titled, '**On the Effectiveness of Private Information Retrieval Protocols**' is my own work and has not been submitted previously by me for taking any degree from Capital University of Science and Technology, Islamabad or anywhere else in the country/ world.

At any time, if my statement is found to be incorrect even after my graduation, the University has the right to withdraw my PhD Degree.

[illegible]

(**Mr. Rafiullah Khan**)

Dated: June, 2020

Registration No : PC133006

PLAGIARISM UNDERTAKING

I solemnly declare that research work presented in the thesis titled "**On the Effectiveness of Private Information Retrieval Protocols**" is solely my research work with no significant contribution from any other person. Small contribution/ help wherever taken has been duly acknowledged and that complete thesis has been written by me.

I understand the zero tolerance policy of the HEC and Capital University of Science and Technology towards plagiarism. Therefore, I as an author of the above titled thesis declare that no portion of my thesis has been plagiarized and any material used as reference is properly referred/ cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of PhD Degree, the University reserves the right to withdraw/ revoke my PhD degree and that HEC and the University have the right to publish my name on the HEC/ University Website on which names of students are placed who submitted plagiarized thesis.

[illegible]

(**Mr. Rafiullah Khan**)

Dated: June, 2020 Registration No : PC133006

List of Publications

It is certified that following publication(s) have been made out of the research work that has been carried out for this thesis:-

Journal Papers

1. **R. Khan**, M. A. Islam, M. Ullah, M. Aleem and M. A. Iqbal, "Privacy Exposure Measure: A Privacy-Preserving Technique for Health-Related Web Search", *Journal of Medical Imaging and Health Informatics.*, vol. 9, no. 6, pp. 1196-1204, 2019.

2. **R. Khan**, A. Ahmad, A. O. Alsayed, M. Binsawad, M. Ullah and M.A. Islam, QuPiD Attack: Machine Learning-Based Privacy Quantification Mechanism for PIR Protocols in Health-Related Web Search, *Scientific Programming*, 2020. DOI: https://doi.org/10.1155/2020/8868686

Conference Papers

1. **R. Khan**, M. A. Islam and M. Ullah, "Quantification of PIR protocols privacy." In 2017 International Conference on Communication, Computing and Digital Systems (C-CODE), pp. 90-95. IEEE, 2017.

2. **R. Khan**, "De-Anonymizing User of Web and Social Network", In 2016 Fourth International Conference on Biological and Computer Sciences (C-BICS 2016), 2016.

3. **R. Khan**, M. A. Islam and M. Ullah, "Revealing PIR protocols protected users." In 2016 Sixth International Conference on Innovative Computing Technology (INTECH), pp. 535-541. IEEE, 2016.

Rafiullah Khan

(Registration No. PC 133006)

Acknowledgements

In the name of Allah, the Most Gracious and the Most Merciful Alhamdulillah, all praises to Allah for the strengths and His blessing in completing this thesis. Peace and Blessing be upon the lovely prophet Mohammed (SALLALLAHOU ALAYHE WASALLAM).

Special appreciation goes to my supervisor, Dr. Muhammad Arshad Islam, for his supervision and constant support. I would like to express my appreciation to the Dean, Faculty of Computing, Prof. Dr. Muhammad Abdul Qadir, Head of Department of Computer Science, Prof. Dr. Nayyer Masood and other faculty members for the motivation, guidance, support, and encouragement.

I am grateful to Jon Kågström the founder of uClassify web services and his team to support me in this research by upgrading my uClassify account to high volume account and their technical support.

I would like to thank my family specially my mother, my wife, and my sisters who always played a very supportive role, boost me up whenever I feel down. I have profound gratitude for them who had to face my educational stress. I am thankful to my friends Mohib Ullah and Dr. Asfandyar Khan and my boss Mr. Muhammad Shahid, who was always there to take me out of stress and motivated me many times. They had more confidence in me than I had in myself. I would like to thank my teachers especially my mentor Syed Rafaqat Ali, for encouraging me to always take care of myself mentally, physically, spiritually, and emotionally.

Last but not least, I would like to thank my teacher, my tutor, my protector, and my best friend, my father Dr. Shafiullah Khan. I am at a loss for words when I try to express what you mean to me. I couldn't have wished for a better mentor. Thank you very much.

Abstract

Due to the exponential growth of information on the Internet, Web Search Engines (WSEs) have become indispensable for the effective retrieval of information. In order to provide the results most relevant to the user, WSEs store the users' profile that may contain sensitive information including the users age, gender, health condition, personal interests, religious or political affiliation, etc. However, this raises serious concerns for the privacy of the user since the identity of a user may get exposed and misused by third parties. To address the issue of privacy infringement while using WSE, researchers have proposed several techniques such as anonymizing networks, profile obfuscation, private information retrieval (PIR) protocols. In the anonymizing network, the user's query is forwarded to the WSE through a chain of routers. In profile obfuscation technique, fake queries are forwarded with the user's query in order to mislead the WSE. While one well-known solution to preserve privacy is a Private Information Retrieval (PIR) protocol called Useless User Profile (UUP) that issues the queries via Peer nodes, thereby hiding user's identity from the WSE. Despite the fact that the aforementioned methods improve the user privacy, yet some previous studies using a machine learning algorithm and user profile show that an adverse WSE is able to break profile obfuscation and anonymizing network methods. However, it is not clear if an adverse WSE is able to break UUP using machine learning techniques.

This thesis investigates the protection offered by UUP. To evaluate the effectiveness of UUP in privacy protection, we propose QuPiD (Query Profile Distance) Attack. QuPiD Attack is a machine learning based attack that determines the distance between the user's Profile (web search history) and upcoming query using a novel feature vector. The proposed feature vector is composed of a set of numeric values of 10 major topics acquired from uClassify service. The results show that the proposed QuPiD attack associates more than 40% queries to the correct user with a precision of over 70%. Moreover, during the investigations, the proposed QuPiD attack behave unexpectedly in some cases, affecting its *precision* and *recall*. Upon detail investigation, three reasons are found responsible for that behavior: (i) variable similarity score between incoming query and user profile, (ii) lack of traces of incoming query in the user profile, and (iii) presence of more than one matching user profiles. We call this behavior as ProQSim (Profile Query Similarity) Effect.

Based on the results, it is concluded that UUP does not provide satisfactory protection to users. We, therefore, develop PEM (Privacy Exposure Measure), a technique that minimizes the privacy exposure of a user while using the PIR protocols. PEM assesses the similarity between the user's profile and query before posting to WSE and assists the user to avoid further privacy exposure. The privacy evaluation experiments with PEM (Privacy Exposure Measure) demonstrates that a user profile created with a web search engine through PEM is 95.97% different as compare to the usual user's profile and thus offers more privacy to the user even in the case of machine-learning attack for $mpeT = 10\%$. (Maximum privacy exposure ($mpeT$) is the threshold value set by the user for exposure of his/her profile to the web search engine). We have established empirically that our proposed privacy protection mechanisms (PEM) significantly improves performance with regard to preserving privacy. Finally, we conclude this thesis with our perspectives and point out some key future research issues in the areas of PIR protocols, adverse models, and our suggested module PEM.

Contents

List of Figures

List of Tables

Abbreviations

AOL	America Online
DCA	Dummy Classification Algorithm
DISPEP	Disjunctive Plain-text Equivalence Proof
DMOZ	Directory Mozilla
GooPIR	Google Private Information Retrieval
NISPP	Noise Injection for Search Privacy Protection
OAS	Optimized Arbitrary Size
ODP	Open Directory Project
OQF	Optimized Query Forgery
P2P	Peer to Peer
PDS	Plausibly Deniable Search
PEAS	Private Efficient Accurate web Search
PEL	Profile Exposure Level
PEM	Privacy Exposure Measure
PEP	Plain-text Equivalence Proof
PFA	Profile Filtering Algorithm
PIR	Private Information Retrieval
PRAW	Privacy-Aware Web
ProQSim	Profile to Query Similarity
QuPiD	Query to Profile Distance
SIGKDD	Special Interest Group on Knowledge Discovery and Data Mining
TF/IDF	Term Frequency / Inverse Document Frequency
TMN	Track Me Not
TOR	The Onion Router

UPIR	User Private Information Retrieval
UUP	Useless User Profile
VPN	Virtual Private Network
WSE	Web Search Engine

Symbols

$Swin$	Session Window
$GSwin$	Group of Session Windows
Lu	Expected user label
$mpeT$	Maximum privacy exposure threshold
P_{model}	Classification Model
Pqi	User ith query from profile
PU, UP	User Profile
PUv, UPv	User Profile with feature vector values
q	Unclassified query
Q	User query
QoI	Query of Interest
$qsim$	Query similarity score
qv	query with feature vector value
Qv	User query with feature vector
qv	Unclassified query with feature vector
UoI	User of Interest

"Privacy isn't about hiding something. It's about being able to control how we present ourselves to the world."

— Bruce Schneier (*Cryptography and Security Specialist*)

"Privacy is not an option, and it shouldn't be the price we accept for just getting on the Internet."

— Gary Kovacs (*CEO, AVG Tech*)

"We don't need you to type at all. We know where you are. We know where you've been. We can more or less know what you're thinking about."

— Eric Schmidt (*CEO, Google Inc*)

Chapter 1

Introduction

1.1 Introduction and Motivation

Online users' privacy is a delicate issue that has been unfortunately largely overlooked by technology corporations and especially general public since from the birth of the Internet. Many online businesses and services such as web search engines (WSE), online retailers and social network sites exploit the user's personal data for profit [1]. This collection of personal data possessed by these companies are often used for targeted advertisements and marketing [2]. Although these companies claimed that they maintain the user's profile in order to deliver them relevant and personalized results. However, the indiscriminate collection and exploitation of users' information without their knowledge is an indecent act. Worryingly, the terms and conditions of these online services are intentionally kept vague as a precautionary measure. Similarly, when it comes to web search engines, the issue of the user's privacy becomes more critical and complex. Such as the Google's current terms and conditions which narrate that they got the license to store, reproduce, modify and use the contents uploaded, submitted, stored, sent or received by the user through their service [3]. For that reason, many online services introduced user privacy setting options. However, a study on Facebook shows that these privacy setting cannot offer protection to the user's privacy in practice due to either complex privacy setting or lack of desired options [4]. Moreover, another study found that the privacy policies of online service providers are usually not aligned with the user's privacy values [5]. This matter became even worse after President Trump signed the legislation enabling internet service providers to sell user data without their consent [6]. However, the European Union took this matter more seriously by passing laws regarding users' privacy [7]. The Court of Justice of the European Union (CJEU) even forced Google to introduce the *"right to be forgotten"* options [8, 9]. In 2016, the European Union has revamped the data protection laws and bound Web service providers to enforce third parties that a person wants his/her information to be deleted [10]. However, these privacy rules and regulation are applicable in Europe only and Web service providers chose not to generalize for them globally [11, 12].

With the beginning of World Wide Web services, diverse kind of computer manageable data is pouring in from various sources, whereby recent advancements in the communication networks facilitate users to access and share a large volume of information in a minimum amount of time [13]. The service starting based on the concept of a global information system, has now became a huge pile of information of almost every sort [14]. In order to find relevant and specific information over the internet efficiently, web search engines were introduced in 90's [15]. Before the invention of Web Search Engines, it was an enormous challenge for internet users to find the relative and topic-specific information in the globally scattered data sources. First ever web search engine was introduced in September 1993 with the name W3Catalog [15]. Presently, the most popular web search engines are Google, Bing, and AOL, etc. According to Pew Internet & American Life Project report 2012, 91% of users always find correct information when searching in the net. The same survey claimed that the use of search engines increased by 6% during the last decade (between 2002 and 2012) which was 85% and 91%, respectively.

Due to the exponential growth of information on the World Wide Web, Web Search Engines have become indispensable for the efficient and accurate retrieval of information. Considering this fact, web search engines usually collect a wide range of information about the user. Maintaining a user profile is very helpful for ranking relevant search results according to the user's interests and past history. It is also helpful for recommendation systems provided by many websites and used to remember customized user settings. However, the user profile may contain personal and sensitive data which may cause critical privacy breaches. While using WSE, users often do not understand that they expose themselves to a massive leak of information. This issue of users' privacy breach received significant attention in 2005 when the US department of Justice compelled Google to submit records of users' queries [16]. Later America Online (AOL) released (pseudo-anonymized) 20 million queries of more than 650,000 users submitted in three months of time without user consent for research purpose [17]. Although they replace the users' identifiers (e.g. email id, name, IP address, etc.) with fictitious ID, the New York Times was able to deduce some of the users correctly based on their search query patterns [18]. Apart from the scandal, this release of data enabled researchers to explore numerous important areas such as user behavior analysis and prediction [19, 20], query reformulation strategies [21, 22], web user privacy analysis [23], etc. Similarly, the focus of some studies was to extract the explicit and implicit personal information of the users from queries [18, 23]. For example, the New York Times managed to identify the person behind the user # 4417749. User # 4417749 was 62-year-old single women Thelma Arnold living in Lilburn, Georgia with her three dogs [18, 24]. Explicit personal information contains facts that can reveal user identity or behavior. For example, in the query *"Jarrett arnold eugene oregon"* sent by Thelma Arnold, we understand that she is searching about one member of her family in Oregon State. Implicit information is extracted from the whole user profile using advanced methods (such as data mining, etc.) [23, 25]. For example, we can deduce the user's age, gender, the standard of living, wealth, location, sexual orientation, political and religious affiliation, health condition, etc.

There is a misconception among the people about the term "Privacy". Usually, people think that privacy is the ability of an individual to isolate themselves or it's a person right to control access to her personal information. Researchers usually argue that disclosing users' personal information is not a problem as they have nothing to hide. However, according to Solove [26], privacy is not just about the

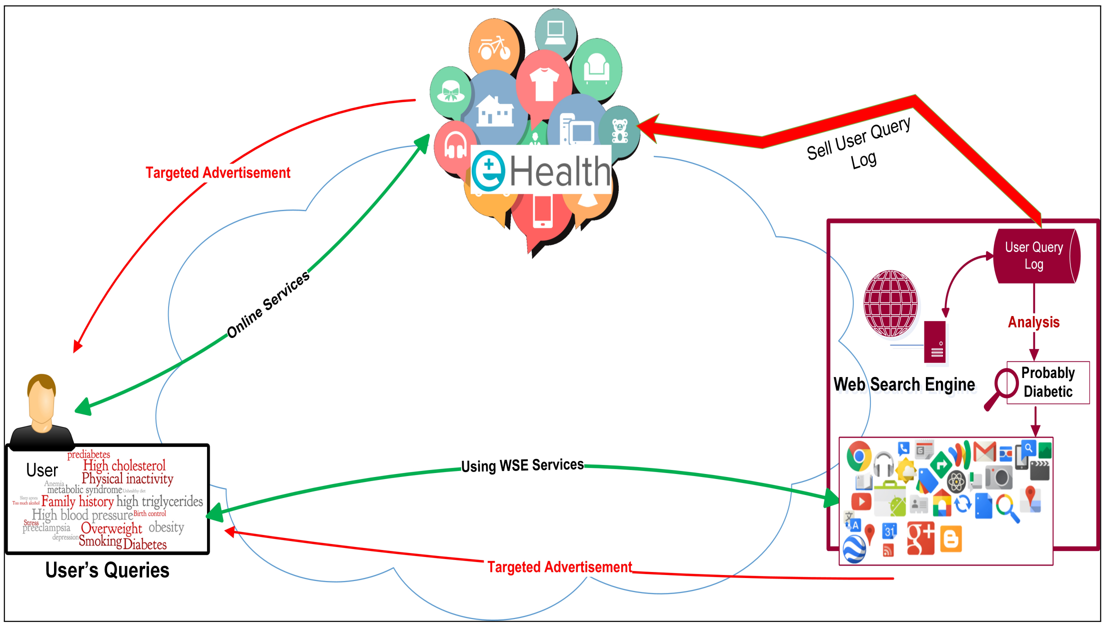


FIGURE 1.1: Web Search Engine collects User's queries for Selling and Targeted Advertisements. [28]

revelation of secret information but it also includes the exploitation of user's personal information. Similarly, Schneier advised users to take this matter seriously as the exploitation of personal information may lead to disastrous consequences [27]. For example, in the context of a Web Search, an Adverse Web Search Engine can disclose or sell user personal queries (such as health and wealth-related queries) to the insurance companies, banks, and advertising agencies (see Fig. 1.1). Fig. 1.1 shows that the user is using different services provided by the web search engine (such as search engine, map, smart phone apps etc.) and most of his/her queries are related to the condition of diabetes. Based on the analysis user's activities in different Web Search Engine services, Web Search Engine infers that the user is probable a diabetic. Web Search Engine then may use this inferred information for targeted advertisements and sell it to other interested companies.

As many popular and commercial web search engines lack dedicated privacy preserving features to ensure user's privacy, alternative search engines have emerged such as DuckDuckGo[1] , Qwant[2] , Start-Page[3] , etc. The aforementioned search engines do not provide any specific mechanism for privacy preservation, instead,

[1]https://duckduckgo.com
[2]https://www.qwant.com/?l=en
[3]https://www.startpage.com

they claim that they do not maintain user profiles and do not collect users' personal information. According to their terms and conditions, they affirm that they only collect search queries without identifying information. However, as could be seen in the AOL scandal, that even an anonymized log cannot provide sufficient privacy to the users [23]. Besides, these search engines cannot be trusted as their implementation is not publicly available.

To overcome these issues of privacy infringement, researchers have proposed several techniques and alternatives. Using these schemes, users can query the web search engine without revealing the query content. However, these solutions rely on cryptographic algorithms which increases the computation cost for both users and service provider. Moreover, web search service providers have no interest to deploy such schemes as their business model is based on the exploitation of personal data. As the deployment of these solutions seems unrealistic, the focus of this thesis is client-side solutions for privacy-preserving web search.

As popular commercial web search engines are not interested in providing privacy preservation mechanisms. Therefore, researchers have proposed various client-side solutions for privacy-preserving web search. These solutions can be classified into four major classes i.e., user anonymizing networks, profile obfuscation, private information retrieval (PIR) protocols, and hybrid techniques. In the anonymizing network (e.g., RAC [29], Onion Routing [30], Dissent [31, 32], and Tor [33], etc.), the user's query is forwarded to the web search engine through a chain of routers for anonymous communication. They are also referred to as unlinkability solutions or anonymous communication protocols [2] as they attempt to hide the user's identity. In the profile obfuscation technique (e.g. TrackMeNot [34, 35], GooPIR [36], etc.), fake queries are forwarded with the user's queries in order to mislead the web search engine. These techniques are also referred to as indistinguishability solutions and their main aim is to alter the user's search pattern [2]. In PIR protocols (e.g. UUP [37–40], UPIR [41, 42], Crowds [43]), a group of users exchange their queries with each other and submit it to the web search engine in order to preserve the privacy, while hybrid techniques (e.g., PEAS [44], X-Search [28]) are a combination of more than one aforementioned techniques.

All the aforementioned proposed techniques for privacy-preserved web-search have some advantages and disadvantages. For instance, user anonymization techniques can offer protection on the network layer by hiding user's network layer identifiers, however, web search engines maintain users profile based on various other identifiers such as cookies and device fingerprints, etc. Moreover, user anonymization techniques are feeble to machine learning attacks [45, 46]. Similarly, profile obfuscation techniques offer relatively better privacy by misleading the web search engine about users' intentions and interests. In these techniques, however generation of realistic fake queries is a challenging task. As machine learning algorithms can easily identify machine-generated queries [14]. The Private Information Retrieval (PIR) protocols offer even more privacy by submitting user's query to the web search engine through other trusted user and can avoid problems related to fake query and profiling [47].

1.2 Research Gap and Problem Statement

Machine learning based attacks have been proven to be the most effective and successful attacks for all privacy major preservation techniques. Machine learning based attacks need user's profile (web search history) and classification algorithm to build a prediction model for the classification of upcoming queries. There are several studies available that evaluate the effectiveness of privacy preservation techniques using machine learning attack. In these studies [2, 14, 45, 46], machine learning attacks are used to classify fake and real queries in case of profile obfuscation techniques or identification of the query originator node in case of the anonymizing network. However, to the best of our knowledge PIR protocols have never been evaluated using machine learning attacks.

Apart from the fact that PIR protocols are bit expensive in terms of time and hardware resources as compere to the profile obfuscation and user anonymization techniques, yet they are considered to be the most secure privacy-persevering web search techniques [47]. This is due to the fact that these protocols do not rely on fake machine generated queries or unknown proxy users. Instead these protocols works with legitimate users and their legitimate queries. However, this is also a noticeable fact that the effectiveness of these protocols in terms of privacy have never been tested against machine learning attacks to the best of our knowledge. The major aim of machine learning attack is to associate the queries to their original users in an anonymized log. Therefore, the performance evaluation and effectiveness of PIR protocols against machine learning attack and improvement in the existing protocols regarding attack is a researchable issue.

The aim of this thesis is to evaluate the effectiveness of the PIR protocols in terms of privacy using machine learning attacks and to propose an improvement in the existing protocol for better protection. For the evaluation of privacy, web search engines are assumed to be an entity whose goal is to work against the privacy-preserving solution and to identify the User of Interest (UoI) queries for profiling purposes. It is assumed that a web search engine is equipped with the user's search history. The user profile contains queries submitted by the user in the past without using any PIR protocol.

1.3 Research Objectives and Questions

Based on available literature, it is clear that most of the popular solutions for privacy-preserving web search (e.g., anonymizing networks and profile obfuscation techniques) are vulnerable to machine learning attacks. However, to the best of our knowledge PIR protocols have never been evaluated using machine learning attacks. Therefore, the major issues addressed in this thesis are related to the evaluation of PIR protocols under machine learning attacks. Moreover, it is an obvious fact that the dynamics of PIR protocols are different from other privacy-preserving web search techniques. For instance, PIR protocols use a group of users who have variable querying behaviors and frequency. Similarly, the size of the user profile (for building prediction model) also varies due to different user behavior. Therefore, in order to test the capabilities of our proposed machine learning attacks from various angles, we address the following research questions in this thesis:

1. **What are the effects of machine learning attacks on the privacy of Private Information Retrieval protocols?**

   (a) *To what extent Private Information Retrieval protocols can successfully protect user's privacy in case of machine learning attack?*

   (b) *Which machine learning algorithm is more suitable for successful machine learning attack?*

   (c) *What will be the impact of profile size on the results?*

   (d) *What will be the impact of the session window size on the results?*

   (e) *What will be the impact of different feature vectors on the results?*

   (f) *Identification of factors that can influence the success rate of machine learning attacks.*

2. **How can we improve the effectiveness and performance of the Private Information Retrieval protocols in the presence of Machine Learning Attack?**

   (a) *Which similarity measure is more suitable for the proposed improvement of the Private Information Retrieval protocols in order to minimize the effect of machine learning attack?*

1.4 Contributions

Regarding the first research question, the baseline was established in previous studies by Peddinti and Saxena [14, 45, 48] and Petit et al. [2, 44, 46]. However, the targets of these studies were anonymizing networks (Tor, GooPIR) and profile obfuscation schemes (TMN) since the dynamics of the PIR protocols are different from anonymizing networks and profile obfuscation techniques, and additionally the machine learning attacks proposed for these solutions cannot be adopted directly. For these reasons, we designed a novel attack, the QuPiD Attack (Query Profile Distance Attack), which is a machine learning based attack that evaluates the effectiveness of the Private Information Retrieval protocol in privacy protection. The QuPiD Attack determines the distance between the user's Profile (web search history) and upcoming query using a novel feature vector. The results showed that the proposed QuPiD attack is successfully able to break the users' privacy, and thus showed that PIR protocols cannot provide satisfactory protection to the users.

Regarding the sub-questions of Research Question 1 (Chapter 3), we have identified some factors that can influence the overall capability of the proposed attack. Although the proposed attack bears better results in breaking the users' privacy provided by PIR protocols. We identified an interesting behavior of the proposed attack that predominantly affects the rate of *recall* and *precision* in some cases. Upon detailed investigation, it is found that this behavior occurs due to three reasons: (i) variable similarity score between incoming query and user profile (history), (ii) lack of traces of incoming query in the user profile, and (iii) presence of more than one almost similar user profiles. We named this behavior the ProQSim effect (Profile Query Similarity).

To answer the Research Question 2 (Chapter 4), we propose the PEM (Privacy Exposure Measure) step as an improvement in PIR protocols. Privacy exposure measure is a privacy estimation mechanism that assesses the similarity between the user's profile and query before posting to WSE assisting the user to avoid further privacy exposure. According to the mechanism of a privacy exposure measure, the user is initially asked to set the privacy exposure threshold. After setting the privacy exposure threshold, the similarity of every user query is then calculated against the user profile. If the similarity between the user profile and the query is above the threshold, the user is asked to reconsider the query. As a consequence, the low similarity between queries and users profile prevents the machine learning attack to expose user's privacy accurately. The Performance of PEM is evaluated in terms of Profile Similarity, Cross-Entropy loss and Kullback-Liebler (KL) divergence.

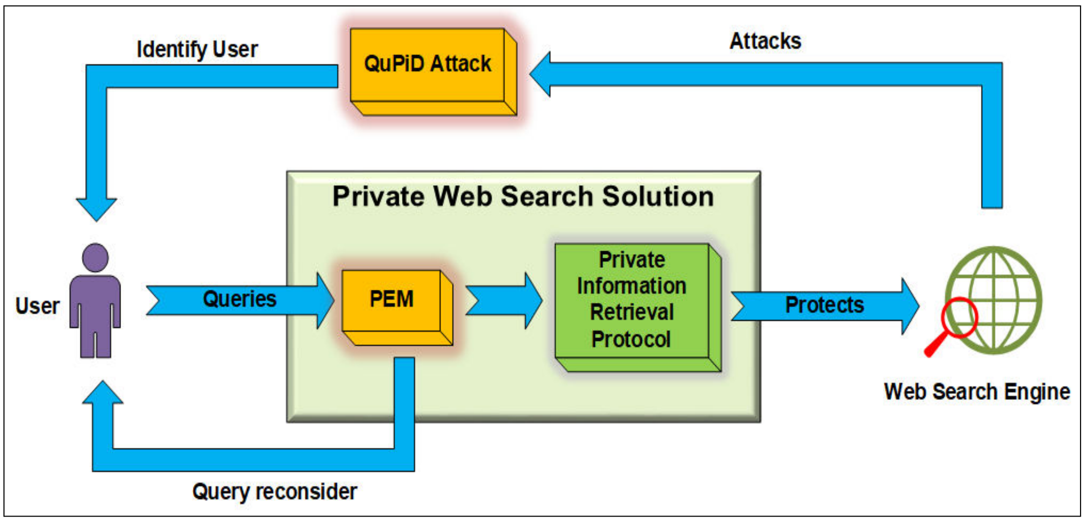


FIGURE 1.2: Our contributions presented in a single scenario (Shown in Yellow color).

We summarize our two major contributions presented in this thesis through the scenario presented in Fig. 1.2 (Shown in yellow color). A user sends a query through private information retrieval protocols. Our proposed step PEM (Privacy Exposure Measure) assesses the profile exposure. If the exposure is more than the threshold value, the query is sent back to the user for reconsideration. Otherwise, the query is forwarded to the web search engine. Finally, the web search engine (adverse entity) uses QuPiD attack to expose the user privacy provided by the private web search solution.

1.5 Thesis Organization

The remainder of the thesis is organized as follows: Chapter 2 contains a review of the state of the privacy-preserved web search techniques. Chapter 3 and Chapter 4 describe the major contributions developed in this thesis and Chapter 5 presents the conclusion of thesis and future work.

A more detail overview of the content of each chapter follows:

1. Chapter 2 is divided into four major sections. In the first section, we discussed all the available private web search solutions that are reported in the

literature along with their shortcomings. In the second section, we discussed all the available attacks and adverse models proposed for the evolution of private web search solutions. In the next two sections, we discussed metrics used to calculate the effectiveness of private web search solutions and query categorization techniques.

2. In Chapter 3, we present QuPiD attack in order to answers the Research Question 1 and its subparts. The QuPiD attack is a machine learning based attack that is used to evaluate the effectiveness of PIR protocols in terms of privacy protection. QuPiD Attack determines the distance between the user's Profile (web search history) and upcoming query using a novel feature vector. We also discuss the design and working of QuPiD attack along with experimental results in detail. Moreover, Chapter 3 also contains a discussion about ProQSim affect its causative factors and results of related experiments. ProQSim effect is an effect that can influence the performance of the proposed QuPiD attack.

3. In Chapter 4, we present PEM (Privacy Exposure Measure) in order to answer the Research Question 2 and its sub-part. PEM is a privacy exposure minimizing technique for Private Web Search. PEM facilitates users to control their privacy exposure while using PIR protocols. PEM assesses the similarity between the user's profile and query before posting to the Web Search engine and assists the user in avoiding privacy exposure. In this chapter, we also discuss the design and working of PEM along with experimental results in detail.

4. In the last chapter (Chapter 5, we summarize and discuss the major contributions of our thesis in detail. We also gave possible future directions related to our contributions.

Chapter 2

State of the Art

Online user's privacy is a delicate issue from the user's point of view. Unfortunately for web search engines, user privacy is just a behavior tracking and used for multiple purposes including profit. To address the issue of privacy infringement while using WSE, researchers have proposed several techniques and alternatives. This chapter presents an overview of the existing private information retrieval solutions along with adversarial models and privacy attacks in detail. We also discuss the methods through which their performance was assessed in terms of privacy protection and performance. Moreover, this section also contains a discussion about different query categorization techniques.

2.1 Private Web Search and Solutions

Solutions for private web retrieval can be categorized into four major categories (i) systems that use Private Information Retrieval Protocols to ensure privacy (ii) systems that try to contaminate the user profile or history using indistinguishability techniques (iii) systems that break the binding between query and user using unlinkability techniques, and (iv) Hybrid systems. As the focus of this research is Private Information Retrieval Protocols, we will discuss Private Information Retrieval Protocols in detail as compared to the rest of the solutions.

2.1.1 Private Information Retrieval Protocols

The concept of private information retrieval narrates that user can get information from the source without revealing the retrieved item [49, 50]. Normally this facility should be offered by the web search engine and many web search engines already claim to provide this facility[1][2][3][4]. In this regard, Pang et al. [51] proposed web search engines oriented user privacy preservation. The authors suggested that the query should be decomposed into multiple buckets of terms and each bucket must be encrypted with some homomorphic encryption algorithm. The homomorphic property states that the product of two encrypted text $Enc(a)$ and $Enc(b)$ is equal to the encryption of the product of two plain-texts $Enc(a \times b)$ (shown in Equation 2.1).

$$Enc(a) \times Enc(b) = Enc(a \times b) \tag{2.1}$$

The WSE will provide the relevance score each each encrypted bucket (without knowing the content as it is encrypted). In the end, the user will de-crypt the result and find the relevant documents based on the score. However, in terms of time complexity, homomorphic algorithms are costly [52]. Moreover, many prevailing web search engines collect user information and use it for various purposes. Therefore the problem of getting information privately must be dealt with at the user side. Therefore, in order to provide anonymity to the user, the first ever scheme proposed was Private Information Retrieval (PIR) by Chor et al. in 1998 [49, 50]. This scheme was proposed to get the data from the replicated copies of the database. Their solution was to break the user's query into small select queries and send all queries to different copies of the same database and gather results of all queries. Thus, the user intention will remain secret. However, the computational cost of the solution was too high [41]. Additionally, this solution was proposed for SQL databases.

[1]www.duckduckgo.com
[2]www.startpage.com
[3]www.ixquick.com
[4]www.yippy.org

According to Pfitzmann and Waidner [53], three types of anonymity can be provided in anonymous communication: sender anonymity, receiver anonymity, and unlinkability between sender and receiver. In the first case, the sender will remain hidden during communication, in the second case the receiver will remain hidden while in third case both the sender and receiver can be identified. However, in the latter case they cannot be identified as communicating with each other. Therefore, in light of Pfitzmann and Waidner explanation of private information retrieval (PIR), the system must provide sender anonymity while the rest can be provided as needed. Similarly, according to Petit et al. [2], a PIR protocol must have the following three algorithms: (i) query construction and protection (using encryption at least), (ii) retrieval of information privately, and (iii) reconstruction of results for the user.

The Peer-to-Peer solution uses a complete decentralized architecture to hide user identity. In Peer-to-Peer schemes [38–40, 54–58], users who wanted to submit queries privately create a collaborative group and submit the queries on behalf of each other. These solutions are often called user-based solutions [2] as they use collaborative users. There are many techniques available to users to collaborate and get information privately. Some techniques use shared memory locations, while some use encrypted query exchange and other methods. These methods can provide better privacy, but the response time of the bigger group is worse [2]. This section contains discussion and working of all available PIR protocols.

2.1.1.1 Crowds

Reiter and Rubin gave peer to peer web PIR concept [43] in 1998, and they used public peers to ensure privacy in web transactions. Their approach was based on the idea "blend into a crowd" and for that reason, they called their system the "Crowds". In addition to the scheme, authors also proposed a scale for anonymity, which provides different degrees of anonymity ranging from "Absolute Privacy" to "Provably Exposed". In the proposed scheme, the crowd is the collection of users who want to perform web transactions anonymously. A user can join the

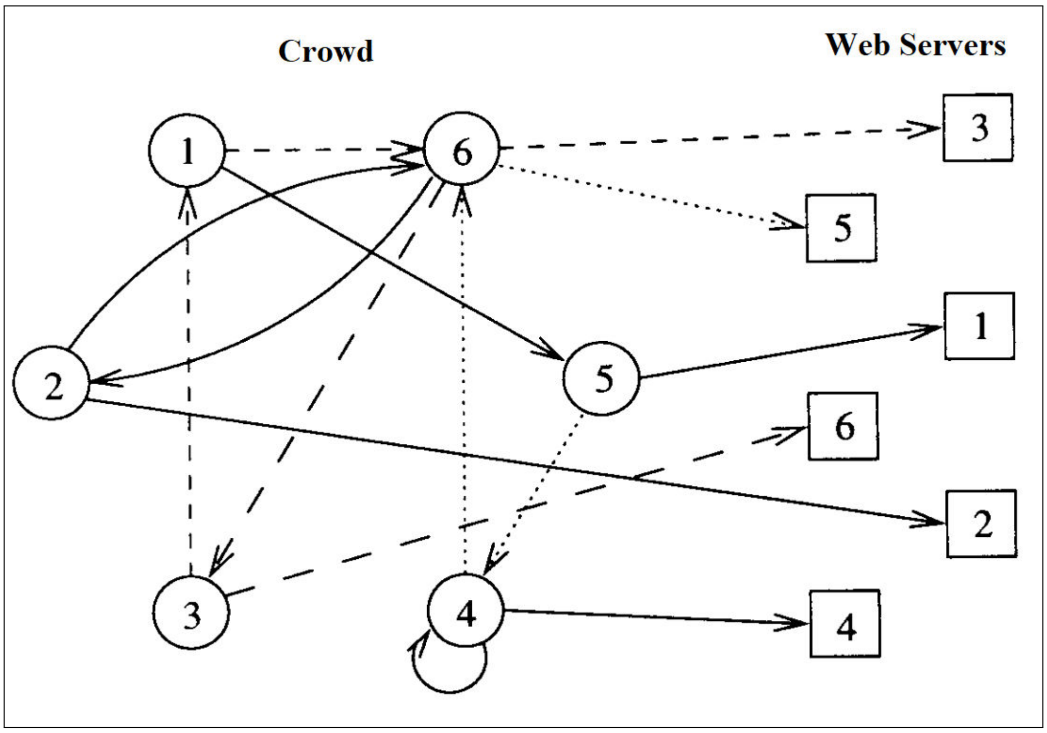


Figure 2.1: Paths in a Crowds. [43]

process by starting the application named "*Jondo*". Once "*Jondo*" is started on the user's computer, it contacts the server, which they named "blender". The "blender" keeps the record of all online clients (who use "*Jondo*"). After that, the user will set *Jondo* as default proxy server and all requests will be sent directly to *Jondo*. After this phase, the system is ready for anonymous web transactions. Every request a user has made is send to the other active *Jondo* by picking a random member in the Crowd *Rm*. The *Rm's Jondo* will run a biased coin flip-a-coin algorithm to decide whether to forward that request to the web server or forward it to another *Jondo*. During this whole process, each *Jondo* will keep the record of all requests it forwarded which record is then used for the reply phase. The flip-a-coin algorithm is very effective in preserving user's privacy, as the intermediate user of the communication was aware of the content but not the identity of the requester. They have also characterized the user's anonymity properties using *degree of anonymity* scale. The paths of the requests in "Crowds" are shown in Fig. 2.1.

Wright et al. [59] proposed a predecessor attack to find weaknesses in Crowds. They noted that the Crowds system stood weakly against it. In that attack, a number of attacking nodes join the Crowds and record all the queries which passed through them. After getting a large number of entries, the attackers simply tally their log and find the victim. Similarly, Viejo et al. [40] reported that a lengthy path, blender, and encryption and decryption at each node will introduce a notable delay in the overall procedure. Moreover, as the system acts as a complete graph, each edge will need a separate symmetric key which will be greater in number. From the user point of view, the profile of the group for optimized search can only be possible if group members share the same interests and if a user has to hide his/her own information as Crowds only protects the transport of the data. Viejo et al. [40] also proposed a solution to deal with the nodes that refused to forward any query (selfish node).

2.1.1.2 Useless User Profile Protocol

Castella-Roca et al. proposed the Useless User Profile (UUP) [37], a more secure technique than Crowds [43] for private information retrieval. In UUP, they introduced a group concept and used a group encryption for query hiding other than in "Crowds". UUP is composed of four sub-protocols i.e. Group setup, Group key generation, anonymous query retrieval, and query submission and retrieval. The process starts with the group setup step, during which the user Ui sends a request to the central node Cn to join a group. The central node acts as a facilitator and generates groups from the individual users. When a central node receives n requests from n users, it creates a group of n users. In the next step, group keys are generated. Each user generates an ElGamal [60] single public key and also generate a random private key. The common public key is then generated by adding the public keys of all group members. All users in the group encrypt their queries with the common public key and send them to each other. Each user permutes and re-mask his/her query and sends it to the next user. When the last user of the group re-masks the queries, all the ciphered queries in the group are broadcasted in the group whereby each user receives its assigned ciphered query and submits it to the web search engine. As the query requester became anonymous due to multiple re-masking, the results of the query are then broadcasted in the group.

Unlike Crowds the UUP provides more privacy. However, the cryptographic operations introduce about 6.8 seconds delay in the whole process [2]. To minimize the delay, Romero-Tris et al. [39] proposed some modification in the process. They suggested to decrypt queries before the last broadcast and for all users to submit all queries to the search engine including their own as the last step will remove the last step of result broadcast. By introducing these modifications, they reduce the delay from 6.8 seconds to 3.2 seconds [2].

Lindell et al. [47] found that the UUP stood weakly against adversary nodes. They studied the behavior of protocol under various attacks that were launched by adversary nodes present in the group. The attacks include denial of service, replacement attacks, and cryptography attacks. Lindell et al. pointed out that the shuffling mechanism proposed by the UUP, cannot guarantee that users will follow the protocol correctly. They launched four different attacks i.e. "Targeted public-key attack", "Stage-skipping attack", "Input-replacement attack", and "Input-tagging attack" during the private shuffle phase. The result showed that UUP was vulnerable to these attacks. For that reason, Lindell et al. proposed a verification step to ensure that none of the users cheats during shuffling. They suggested an onion-layered two phases encryption (El-Gamal [60] and CCA2-Secure [61]) for query results and symmetric key encryption for data. After the shuffle, each user checks whether his query is present in the encrypted text or not. If found then the user will send true. If all parties send true then it means that no malicious behavior was reported during the shuffle. Although they solved the problem of malicious nodes, Romero-Tris et al. [38] indicate that this improvement introduces more than 6.8 seconds delay (worse than the first version of the UUP).

In order to improve privacy with a minimum delay, Romero-Tris et al. [38] proposed a new protocol, the Multi-party private web search, with untrusted partners (MPWS-UP). The MPWS-UP protocol is also composed of four steps. The first step is a group set-up during which every node wishing to send a query in private, requests the central node for group joining the group. When the central node received n requests, it creates a group. After the group creation, an Optimized Arbitrary Size (OAS) Benes network [62] is used for secure and efficient shuffling.

OAS Benes networks gives permutation of n inputs with 2×2 switches. Romero-Tris et al. used t OAS Benes network where t shows the number of trusted users in the system. For the verification step, they used zero-knowledge proof protocols using PEP [63] (Plain-text Equivalence Proof) and DISPEP [63] (Disjunctive Plaintext Equivalence Proof) based on ElGamel [60]. By improving the shuffling process, they obtained a two times lower delay then the Lindell et al. [47] proposed improvement.

Cao et al. [64] noted that in Lindell et al. [47] the proposed scheme can suffer from the malicious node. They claimed that an adverse node can submit a fake or modified query to the search engine and then broadcast the results of a fake or modified query in the group. Cao et al. also suggested that the user should permute his query with $n-1$ other queries and then submit them in the group. In this way, the malicious user can identify true query with the probability of $\frac{1}{n}$.

Another problem in the Lindell et al. [47] proposed schemes was that although it can detect that some user went malicious, it cannot identify the actual culprit. Corrigan-Gibbs et al. proposed Dissent [31, 32], having the capability to identify the malicious node and provide better protection. Compered to Lindell et al., Dissent used the different shuffling algorithm. First each user encrypts the query Q with the secondary public keys of each member Q^s. Then Q^s is re-encrypted with the primary public key of each user $(Q^s)^p$. After receiving $(Q^s)^p$, the double encrypted queries, they are then sent other group members. Each member deciphers the data with a primary private key, permutes and re-masks the internal ciphered queries and sends it to another group member. Finally, the last user broadcasts the final permuted data to all group members. Each user can verify the correctness of the protocol by finding their own query in the results. Otherwise, the user will start the blame phase and will find the culprit by comparing the messages. With all these improvements in privacy, Mokhtar et al. [29] reported that Dissent does not perform well with a higher number of nodes. One of the major issues in the entire family of these protocols is to find trusted partners and personalized results against the query. To handle this issue, Viejo et al. [56] suggested involving the social network friends of the user to get information privately. They proposed an intermediary approach between privacy and search result personalization. Viejo et al. consider a user U who wants to execute some query q in a secure manner, could exploit the existence of online friends present on the social network. The user U may forward the query q to some online friend, the friend may submit the query to the WSE or forward it to another friend in the circle and the process continues. The working of this scheme is very similar to the UUP protocol, except that for the group creation process that has already been created between the friends, no central node is needed.

Privacy-wise this is more secure, as the user will become anonymous in the circle of friends. Similarly, as friends normally share hobbies and interests, the user can get personalized results.

In continuation of this, Erola et al. [65] proposed an enhanced version of the protocol which assists users to select the final forwarding node based on the measure ”Profile Exposure Level” (PEL). Similarly, they also enable the final forwarding node to accept or reject the query for forwarding based on another measure ”Selfishness Test”. Similarly, Rebollo-Monedero et al. [66] give a mathematical analysis of a situation in which the user part of his/her query through other user and submits the other part to himself/herself. Rebollo-Monedero et al. use a probability distribution to model user behavior and analyze the privacy leakage using Shannon entropy [67].

2.1.1.3 User Private Information Retrieval Protocol

Domingo-Ferrer et al. [41, 42] proposed a shared memory based private information retrieval protocol UPIR. The UPIR considers a group of cooperative users who want to retrieve the information privately by submitting queries on behalf of each other. Unlike Crowds [43] and UUP [38–40, 56], none of the group members contacts another group member, instead they use a shared memory location to submit the query and retrieve the answer. For group creation and handling, a facilitator node was used which they called Cn. The process starts with the group setup step, during which user U sends a request to the central node Cn

to join a group. After the creation of the group, the dealer node creates v keys and distributes those keys among b blocks of size k. Then the dealer sends one block of k keys to each other through a secure manner (using the public key of each user). Once each user has received the keys of one block, the dealer erases the v keys from its memory, where v represents the set of users, b represents the memory spaces, and k represents the memory space consisting of k users. These keys are then used to encrypt the query and query answer. For query submission, any user u_i wanting to submit some query q_i randomly selects one key x_{ij} from the k keys from his block. In the next step user u_i reads the memory sector m_{ij} which is corresponding to the key selected by the user i.e. x_{ij}. After reading the data from the position m_{ij} it is decrypted by x_{ij}. The decryption outcome leads to five cases:

1. If garbage value is retrieved, it means the memory is ready to accept the query. In this case, the use u_i encrypt the query q_i with the key x_{ij} and place it at sector m_{ij}.

2. If some other query q_j is retrieved, it means that some other user u_j placed the query for u_i to submit it on behalf of u_j. In this case user u_i submits the query q_j to the search engine, encrypts the query reply with x_{ij} and places it at the same location m_{ij}. After this procedure, the user u_i start the process again from the first step i.e. key selection.

3. If the answer of some old query q_{jprv} that was submitted by u_j is retrieved, it means that user u_j hasn't read the answer yet (as each user supposed to erase the memory after reading the answer). In this case, user u_i starts the process again from the initial step.

4. If some old query q_{iprv} issued by user u_i is retrieved, it means this query is waiting to be entertained by some user of the group U. In this case, user u_i will have restart the process beginning with step one.

5. If the answer of some old query q_{iprv} issued by u_i is retrieved, it means that this was an old query issued by u_i and its answer has not yet been collected.

In this case, the user u_i will collect the answer, encrypt his new query with the x_{ij} key and will then place it at same memory location m_{ij}.

In order to collect the answer of the query, the user u_i keeps reading the memory location m_{ij} with regular intervals. When the answer of query q_i is found, the user u_i collects the answer and erases the memory location. However, if the answer does not appear in a specified time interval and time out occurs, the u_i selects a new key from the key pool and starts the query submission process again from initial step.

Nevertheless this solution not only hides users but it also makes the user in the group anonymous. However, one of the major drawbacks of this solution is its response time. According to Petit et al. [2], its best and worst time is 5.13 seconds and 30.74 seconds (network latency not included). Similarly, Viejo et al. [56] noted that authors haven't studied the memory space requirements as the protocol must be capable to manage a high volume of information. Moreover, the frequent access of the memory sector by the user will introduce a notable overhead to the network. Romero-Tris et al. [40] also claim that this protocol is vulnerable to a predecessor attack.

Stokes and Bras-Amorós [68] proposed an optimized configuration UPIR protocol. They discussed two main configurations of the protocol i.e. a single memory location (equation 2.2) shared by all the group members [41, 42] and multi-memory location (equation 2.3) for each communication. In a multi-memory location, each user shares a different memory location with every other user and users can split and place their query on memory locations [68]. The configuration of both the scheme are given as follow:

$$SingleMemoryLocation \quad nc = 1, du = 1, dc = nu \tag{2.2}$$

$$MultiMemoryLocation \quad \frac{nu(nu-1)}{2}, du = nu - 1, dc = 2 \tag{2.3}$$

where, nc represents the memory locations, nu represents users, du represents the database accessed by users and dc represents a pair of users who share the common memory location. In an optimal configuration, their proposed main configuration is shown in Equation 2.4:

$$nc\,du = nc\,dc \tag{2.4}$$

With the use of this configuration the following tasks could be achieved: as the number of dc grows, the memory sectors and required keys will decrease providing storage efficiency. As the number of dc increases, the expected waiting time for query submission and retrieval will be decreased providing time efficiency. Similarly privacy-wise, the increase in du will secure the user in front of other users as the query will be distributed to the wider subset of memory. The profile of the specific user will be hidden or diffused in the $du(du-1)$ users, which will provide privacy to the user in front of the web search engine. Stokes and Bras-Amorós mentioned some deficiencies in both schemes, the single memory location (SML) schemes and multi memory location (MML). This means that the WSE will not be able to find the original source of the query, however, every group member knows what query is been posted by whom and group members can find the source node of the query using the ”replacement attack” [47]. Similarly, the multi-memory location scheme is also vulnerable to the ”replacement attack” and is additionally expensive in terms of cost.

One of UPIR's major problems is that it does not have a mechanism to deal with a situation in which all members refuse to submit a query present on the memory location or in which the query is not entertained within a maximum due time. To deal with this problem, Stokes and Bras-Amorós proposed a self-query submission feature in UPIR and provided two new variations of the same protocols [69]. The first protocol configuration deals with the problem of the privacy of the user in front of the server. Stokes and Bras-Amorós defined three terms (i) Real profile, which contains actual queries submitted by the user u, (ii) apparent profile which contains the queries posted by the user u (queries of other users) and (iii) collinear users who are neighbors of user u who submitting the query $N(u)$. Stokes and Bras-Amorós considered a scenario in which the user repeatedly sent requests for rare queries and his requests were entertained very quickly. The user u will be entertained by $N(u)$ each and since the neighbor is limited and known to the server (according to the assumption made in [69]), user u id will never be the part of the rare query that makes it suspect. To solve this problem, they proposed the second variation in the protocol [69]. They changed step two and three of the previous query submission protocol with the following step: "if the memory sector contains a query posted by either the user himself or another user, then u forwards the query to the server, gets the result and writes it on the same memory location after encrypting and restarting the protocol for posting a new query". This process provided an edge to the user u to hide in $\frac{1}{N(u)}$ users provided by all users posting queries with regular intervals. However, in a scenario where user u posts queries with regular intervals and neighbors do not, there are more chances to infer the u real profile from the apparent profile.

Similarly, the third variation of the protocol provided the solution to the problem found in the second variation. In the third variation, only step three of the query submission part is replaced with the following step: "if the memory sector contains the query posted by the user u himself, then u will forward the query to the server with a probability function to decide whether or not to submit his/her own query. If the decision probability is affirmed, then u will send the query to the server, get the result and write it after the encryption. In any case, u will start the protocol again for a new query" [69]. The same shortcoming was exploited by Stinson and Swanson [70], who named the "intersection attack". However, they suggested that if in (v, b, r, k) the configuration $v = r(k-1)+1$ is used, this design will resist the intersection attack. Where v represents, the set of users, b represents the memory spaces, k represents the memory space consisting of k users and r represents the users associated with memory spaces [41, 42, 68–70]. The decision probability factor makes the third variation more secure than the previous one [69].

To avoid the risk of an intersection attack [70], Stinson and Swanson proposed multiple variations of the User Private Information Retrieval (UPIR) protocol [71].

Stinson and Swanson called the new scheme the Pairwise Balanced Design (PBD) protocol. Additionally, they introduced a new parameter in the configuration λ (every pair of the distinct point is in exactly λ blocks and λ is a fixed positive integer). (i) In the first and basic protocol of PBD, users follow the following steps for the combination (v, b, r, k, λ). First user Ui selects him/herself as his/her own proxy and submits the query to the server with the probability $\frac{1}{v}$. Otherwise, user Ui selects some random memory space Sl to which it is associated. Then the user selects any group member Uj to submit his/her query from the memory location Sl. This version affords a luxury to the user to choose the forwarder. This facility was absent in previous protocols. (ii) In this version the first step is different while the remainder of the protocol works the same as the first one. In the first step, Ui selects himself/herself as his/her own proxy with the probability $\frac{1}{v}$ and then randomly selects one of the memory locations Sl from the set of memory r and sends his/her own query to the server. Otherwise, it repeats the second step mentioned before. Stinson and Swanson consider a dynamic Private Information Retrieval (UPIR) scheme in which users can join and leave the scheme. In this case, they proposed an extension to both aforementioned protocols. According to the extension they proposed for both protocols, the user can leave the protocol. In that case, the user will simply be deleted from all the memory locations to which he/she belongs and the keys of those group will be recalculated. However, to add a new user, the protocol will have to fulfill the following prerequisites. It will first find any set of memory spaces whose union contains all users. Then new users will be added into the memory spaces according to the configuration [70, 71].

2.1.2 Indistinguishability Solutions

Indistinguishability techniques are used to distort user interests in such a way that web search engine cannot collect user's accurate profile. The most prominent indistinguishability solutions use machine-generated fake queries techniques to distort a user profile, while the other techniques include user profile splitting, query scrambling or transformation, etc.

TABLE 2.1: Comparison between Peer-to-Peer Private Information Retrieval Solutions.

Protocol	**Group Creation**	**Group Status**	**Central Node**	**Shared Memory**	**Query Shuffling**	**Result Broadcast**	**Self-Query Submission**	**Query Encryption**	**Prone to Attacks**	**Personalized Search Results**
Crowds [43]	S	Static	✓	N.A	✓	✗	✓	✗	✓	❸
UPIR [41, 42]	S	Static	✓	S	✗	✗	✗	✓	✓	❸
UUP [37]	S/M	Dynamic	✓	N.A	✓	✓	✗	✗	✓	❷
UPIR (Optimal Configuration) [68]	S/M	Static	✓	S/M	✗	✗	✗	✓	✓	❷
UUP (Lindell et al.) [47]	S/M	Dynamic	✓	N.A	✓	✓	✗	✓	✓	❷
UUP Social Network [56]	S	Static	✗	N.A	✓	N.A	✓	✗	✓	❹
UPIR (Self-Query) [69]	S/M	Static	✓	S/M	✗	✗	✓	✓	✓	❷
PBD [70]	S/M	Dynamic	✓	S/M	✗	✗	✓	✓	✓	❷
UUP (Untrusted Partner) [38]	S/M	Dynamic	✓	N.A	✓	✓	✗	✓	✓	❷
UUP Social Network Selfish User [65]	S	Static	✗	N.A	✓	N.A	✓	✗	✓	❹
Dissent [31, 32]	S/M	Dynamic	✓	N.A	✓	✓	✗	✓	✓	❷

The ✓ shows the availability of the feature while ✗ shows non-availability of the feature. A qualitative comparison is indicated using the following five symbols: ❶ shows very bad, ❷ shows bad, ❸ shows neutral, ❹ shows good, and ❺ shows very good scores. S = Single; S/M = Single and Multiple; NA = Not Applicable.

Fake query based indistinguishability techniques use fake queries to contaminate the user profile or the history. The user interests are distorted, which results in a misleading user profile with the web search engine. Most of the solutions use machine-generated queries to distort the user profile. However, one of the main challenges for these solutions is the generation of realistic fake queries. It is essential that the fake queries are closely related to the user's interests, otherwise the search engine will be able to filter real queries from user profile [72]. Most fake query based solutions ensure that generated fake queries must be closely related to the user profile. Most prominent fake query based solutions are PRAW (Privacy-Aware Web) [73–75], PDS (Plausibly Deniable Search) [76], GooPIR (Google Private Information Retrieval) [36], TMN (TrackMeNot) [34, 35], and others.

Table 2.2: Comparison of Indistinguishability Solutions for Private Web Search.

	Technique	Fake Queries Source	Scrambling Method	Previous Knowledge for fake queries	Privacy	Performance	Personalized Result	Prone to Attacks
Fake Queries	PRAW [73–75]	Web Page	N.A	✓	❸	❹	❸	✓
	PDS [76]	Web Page	N.A	✓	❸	❹	❸	✓
	Social Media Profiling	Social Network	N.A	✗	❸	❹	❸	✓
	GooPIR [36]	Dictionary	N.A	✗	❸	❺	❷	✓
	TMN [34, 35]	RSS Feeds	N.A	✓	❸	❺	❸	✓
	Viejo et al. [55]	Wordnet, ODP	N.A	✓	❷	❹	❸	✓
Query Transformation	Query Scrambling [79]	N.A	Statistical	✗	❸	❹	❶	✓
	Dispa [80–82]	N.A	Generic Cookies	✓	❸	❹	❹	✓
Profile Obfuscation	Hierarchical User Profile	N.A	Interests Tree	✓	❸	N.A	N.A	✓
	UPS [77]	Wikipedia, ODP	Interests Tree	✗	❸	N.A	N.A	✓

The ✓ shows the availability of the feature while ✗ shows non-availability of the feature. A qualitative comparison is indicated using the following five symbols: ❶ shows very bad, ❷ shows bad, ❸ shows neutral, ❹ shows good, ❺ shows very good scores, and NA = Not Applicable.

Similarly, profile obfuscation techniques aim to give privacy preserved personalized results by sending an obfuscated profile with the user query. These profiles help web search engines to create personalize results. However, these solutions need the cooperation of the web search engine, while query transformation techniques use query scrambling or multiple cookies techniques to preserve user's privacy. The most prominent example of the profile obfuscation technique is UPS [77], while the most prominent example of query transformation techniques are query scrambling [78, 79] and Dispa [80–82]. Table 2.2 shows the summary and comparison between major indistinguishability solutions present in section 2.1.2.

Table 2.3: Comparison of Unlinkability Solutions for Private Web Search.

	Anonymity		Personalized Search Results	Performance	Tolerate Free-Riders	Unlinkability	Prone to Attacks
	Sender	Receiver					
Proxy [89]	✗	✗	❷	❸	✗	✓	✓
VPN [90]	✗	✗	❷	❸	✗	✓	✓
Mixed Network [91]	✗	✗	❷	❷	✗	✓	✓
Web Mixes [92]	✓	✓	❷	❷	✗	✓	✓
Tor [33]	✗	✗	❷	❷	✗	✓	✓
RAC [29]	✓	✓	❷	❶	✓	✓	✓

The ✓ shows the availability of the feature while ✗ shows non-availability of the feature. A qualitative comparison is indicated using the following five symbols: ❶ shows very bad, ❷ shows bad, ❸ shows neutral, ❹ shows good, and ❺ shows very good scores.

2.1.3 Unlinkability Solutions

One kind of solution to keep user privacy intact with regard to adverse and curious search engines, is to break the binding between query and user. There are several ways to track and identify users including the IP address, browsers setting and plugins, device fingerprints, cookies, HTTP headers and others [83, 84]. However, these identifiers are sometimes not sufficient to identify the user uniquely and therefore, these identifiers are often used in combination [46]. These identifiers could also be removed [85–87] or altered [88] to confuse adverse entities. Nevertheless, these techniques could not be applied to network layer addressing schemes [2, 46] and users would still be feeble to machine learning attacks [45]. Moreover, these solutions introduce a notable delay in communication due to cryptographic operations.

There are many techniques available to achieve unlinkability such as proxy service [89] or Virtual Private Network (VPN) service [90]. These services serve as a relay and submit a query on behalf of the user to the search engines and send the results back to the user. This solution provides perfect anonymity thereby protecting the user against the search engine. However, proxy servers maintain their own logs which is an even more serious threat as the user has to trust proxy servers. Similarly, most of these services do not use HTTPS connections which is a huge problem as user conversation is open for attackers and eavesdroppers. Moreover, proxy server may contain malicious malware and they could steal the user's cookies. The most common examples of these sort of services are Mixed Network [91], Web Mixes [92], The Onion Routing (Tor) [33], RAC [29], and others.

Table 2.3 shows the summary and comparison between major unlinkability solutions present in section 2.1.3.

2.1.4 Hybrid Solutions

Besides the indistinguishability and unlinkability techniques, there are some techniques that use the combination of above-mentioned solutions to provide privacy to the user while using a web search engine. In this regard, one of the prominent example is PEAS (Private, Efficient and Accurate web Search), which is proposed by Petit et al. [44]. PEAS took the identity concealment feature from unlinkability and query masking from indistinguishability methods to offer better privacy to the users. Although PEAS provide better user privacy with personalized results. However, the development of PEAS is based on the assumption that both the query issuer node and the query receiver node are non-colluding nodes and that they will never share their information of identity and query content of the user. While, in a non-trusted proxy condition, user privacy cannot be guaranteed. To solve the problem of the non-trusted proxy block, Mokhtar et al. [28] proposed X-Search with a trusted proxy block for safe query execution. They used Software Guard Extension (SGX) [93], an Intel's hardware technology to provide a trusted environment for submitting queries. Table 2.4 shows the summary and comparison between major hybrid solutions present in section 2.1.4.

Table 2.4: Comparison of Hybrid Solutions for Private Web Search.

	Indistinguishability		Unlinkability							
	Fake Queries Source	Previous Knowledge for fake queries	Anonymity Sender	Anonymity Receiver	Tolerate Free-Riders	Trusted Nodes System	Privacy	Performance	Personalized Result	Prone to Attacks
PEAS [44]	Disjoint set of Group Profile	✓	✓	✓	✓	✗	❸	❹	❸	N.T
X-Search [28]	Disjoint set of Group Profile	✓	✓	✓	✓	SGX [93]	❹	❸	❸	N.T

The ✓ shows the availability of the feature, ✗ shows non-availability of the feature. While **N.T** stands for not tested yet. A qualitative comparison is indicated using the following five symbols: ❶ shows very bad, ❷ shows bad, ❸ shows neutral, ❹ shows good, and ❺ shows very good scores.

2.2 Privacy Attacks and Evaluation Models

To evaluate the performance of private web search techniques, many attacks, adverse models and systems are available in the literature. Privacy attacks can be categorized into two major categories based on their working i.e., *active* attacks and *passive* attacks. In *active* attacks, the adversary attempts to breach user privacy on runtime such as DoS attack, Flooding attack and others. In *passive* attacks, adversary launches its attack on query logs. Passive attacks are usually based on machine learning techniques and are often used in adverse evaluations models to evaluate the performance of private web search solutions such as SimAttack, Machine-Learning attack and other. The details of different types of *active* and *passive* attacks are given as follow.

2.2.1 Active Attacks

In a private web search, active attacks are those attacks in which an adversary attempt to breach the user privacy in runtime by altering the working of the privacy protection mechanism. The details of some of the active attacks available in the literature are discussed as follows.

2.2.1.1 Timing Attack

Timing attack [94, 95] was proposed to disclose the user information by measuring the time taken by the user accessing a specific web page. Usually, most of the web browsers use the caching system to reduce the bandwidth and time needed to access needed contents. Therefore, an adversary can exploit the repeatedly requested content for de-anonymization of the user.

2.2.1.2 Congestion Attack

Congestion attack [96] observe the changes in traffic patterns for de-anonymizing the users' identities. This attack uses to identify the Tor [33] relay routers using congestion on various paths followed by Tor users. Once the path of the communication is identified, the user anonymity provided by these schemes is broken.

2.2.1.3 Predecessor Attack

Wright et al. [59] proposed a predecessor attack to find weaknesses in Crowds [43]. Authors noted that Crowds system stood weakly against it. In that attack, a number of attacking nodes join the Crowds and record all the queries which passed through them. After receiving a large number of entries, attackers simply tally their log and find the victim.

2.2.1.4 Flooding Attack

This attack is used to expose the identity of the originator (user) of a certain message in Web Mixes [92] and Crowds solutions [43]. In this attack, an attacker floods the whole anonymity service with messages to reveal the identity of the originator (user) of a certain message.

2.2.2 Passive Attacks

In a private information retrieval, passive attacks are those attacks in which adversary use Machine-Learning techniques over different kinds of server logs to uncover users' identities. These attacks use either unsupervised Machine-Learning algorithms (clustering algorithms such as K-Means [97], Expectation Maximization (EM) Cluster [98]) or supervised Machine-Learning algorithms (classification algorithms such as Logistics Regressions [99], Support Vector Machine (SVM) [100]) to uncover user's identity. The details of some supervised and unsupervised algorithms used for passive attacks are discussed as follows:

2.2.2.1 K-Means

K-Means is an unsupervised Machine-Learning algorithm, use to split a set of observations into K number of clusters using the minimum distance from the centroid of the cluster. More precisely, in a given set of observations $(O_1, O_2, O_3, ...O_n)$, K-Means split this data in k sets $(Set_1, Set_2, Set_3, ...Set_k)$ by calculating the minimum square sum distance from μ_i (the mean of the observations in Set_i or centroid of the cluster) as shown in Equation 2.5. This algorithm is usually used to cluster fake and real queries from the queries set.

$$\sum_{i=1}^{k} \sum_{O \in Set_i} ||O - \mu_i||^2 \tag{2.5}$$

2.2.2.2 Expectation Maximization (EM) Cluster

The Expectation Maximization (EM) clustering algorithm uses the maximum probability feature to cluster observations. EM allocates a probability distribution to each observation showing the probability of it belonging to each of the clusters [98]. EM clustering algorithm usually uses a Gaussian Mixture Model (GMM) to improve the density of a single-density approximation technique using multiple Gaussian probability density functions to model the data. Principally, the EM clustering algorithm uses the following steps to get the optimal model. Initially, for each observation x, the system calculated the membership probability in each cluster (also called an Expectation step). Then in the next step, the probability weights are readjusted and the previous step is repeated (this step is also called a Maximization step). First steps are repeated until the stopping criteria (the third step) is satisfied [101].

2.2.2.3 Logistic Regression

The logistic regression model is usually used for binary classification problem [99]. Logistic regression use probability to classify the observation x to positive class C_p.

$$Pr(Y = C_p|X = \vec{x}) \tag{2.6}$$

Where, Y shows the predicted output variable, Y belongs to either positive class C_p or negative class C_n. While X is the input variable or feature vector. In the simplest form the probability of $\vec{x}$ belonging to positive class C_p is defined as a linear function. However, since linear functions cannot be bounded between one and zero, the logistic function is used to remove range restrictions (shown in Equation 2.7:

$$logit\Big(p(C_p|\vec{x})\Big) = log\left(\frac{p(C_p|\vec{x})}{1 - p(C_p|\vec{x})}\right) \tag{2.7}$$

After mapping the probabilities $\vec{x}$, shown below (Equation 2.8):

$$log\left(\frac{p(C_p|\vec{x})}{1 - p(C_p|\vec{x})}\right) = \theta_o + \vec{x} \cdot \vec{\theta} \tag{2.8}$$

where θ_o shows the error term and shows the regression co-efficient. The resultant probability of $p(C_p|\vec{x})$ is expressed as (Equation 2.9):

$$\left(p(C_p|\vec{x})\right) = \frac{1}{1 + e^{\theta_o + \vec{x}\cdot\vec{\theta}}} \tag{2.9}$$

The event classification can be defined using the threshold value of 0.5 such as if $p(C_p|\vec{x}) \geq 0.5$ then the observation $\vec{x}$ belongs to a positive class.

2.2.2.4 Support Vector Machine

The Support Vector Machine (SVM) is a supervised machine learning algorithm that uses a non-probabilistic binary linear classifier for data classification. In the case of binary classification, SVM computes hyper-plane to separate two classes. More precisely, let us suppose a hyper-plane equation (Equation 2.10):

$$\vec{w} \cdot \vec{x} - b = 0 \tag{2.10}$$

where $\vec{w}$ is normal vector and $\frac{b}{||\vec{w}||}$ shows the distance from its origin along the normal vector $\vec{w}$. However, two classes may not be separable linearly. Therefore Vapnik and Cortes. [100] introduce the concept of soft-margin (Equation 2.11).

$$y_i(\vec{w} \cdot \vec{x} - b) \geq 1 - \varepsilon_i \quad and \ \varepsilon_i \, 0, \forall \in [\![1, n]\!] \tag{2.11}$$

where ε_i shows slack variable (introduced by Vapnik and Cortes. [100]), n shows the number of points and $\vec{x}$ is a data point belonging to the class y_i.

2.2.2.5 Vector Space Model

Vector Space Model (VSM) is a supervised machine learning algorithm that treats text documents as vectors [102]. In VSM, all terms in a document are represented as a vector (Equations 2.12 and 2.13):

$$U_p = [t_{1,a}, t_{2,a}, t_{3,a}, ...t_{n,a}] \tag{2.12}$$

$$Q = [t_{1,b}, t_{2,b}, t_{3,b}, ...t_{n,b}] \tag{2.13}$$

where U_p shows user profile, Q shows query and $t_(x, x)$ shows terms in each document. If a term occurs in a document, its value in the document vector is non-zero. There are different mechanisms available to compute the value such as Term Frequency Inverse Document Frequency (TF-IDF) [103], Boolean Model[104] etc. For similarity calculation between two documents, Cosine similarity is usually used. The Cosine similarity is shown in Equation 2.14.

$$cos\theta = \frac{U_p \cdot Q}{||U_p|| \, ||Q||} \tag{2.14}$$

where U_p. Q is a dot product of two vectors and $||U_p||$ and $||Q||$ represent the norm of a user profile query. The norm is calculated as follow (Equation 2.15):

$$||Q|| = \sqrt{\sum_i^n Q_i^2} \tag{2.15}$$

2.2.2.6 Random Forest

Random Forest is a supervised machine learning algorithm that classifies the data using a multitude decision tree [105]. In a random forest, the decision tree is created from a randomly selected subset of training data. Following it uses votes from different decision trees to classify the test object.

2.2.2.7 Alternating Decision Tree

Alternating Decision Tree (AD-Tree) is a supervised machine learning algorithm that consists of an alternation of decision nodes to classify the data [106]. An AD-Tree consists of decision and prediction nodes. The decision node specifies a base condition, where as the prediction node contains a predicted number. The basic element of AD-Tree is the rule and each rule is consists of three components. These components are precondition, condition and two scores. Where a precondition is a logical conjunction of conditions, and the condition is a predicate and two values show possible outcomes of a base condition. The AD-Tree is the generalization of the voted decision tree, decision tree, and voted decision stumps.

2.2.2.8 Zero Rule

Zero Rule (Zero R) is a rule-based supervised machine learning algorithm that learns or evolve rules and then uses them for classification. It is the simplest classification technique, which relies on the target and ignores all predictors. Usually, Zero R is used as a baseline or benchmark for other classification algorithms. For rule creation, the most frequent value is used [12].

2.2.3 Adverse Evaluation Models

Adverse models are the conceptual design of the systems developed to evaluate the performance of privacy mechanisms. Adverse models are composed of different modules including the attack module, which is usually (not necessarily) based on machine learning techniques. The major purpose of the adverse models is to provide a testing platform to the researchers to evaluate their proposed private web search techniques in terms of privacy. This section contains detail discussion about all available adverse models for the evaluation of various private web search solutions.

2.2.3.1 Peddinti and Saxena Model

Peddinti and Saxena [14, 45, 48] proposed their adverse model to evaluate two popular unlinkability (anonymizing networks) and indistinguishability (profile obfuscation) private web search solutions i.e., Tor [33] and TMN [34, 35, 95]. In their proposed adverse model, WSE is supposed to be an entity whose work is to break the privacy-preserving solutions to identify users and their queries for profiling purpose. They devise a different strategy for both Tor and TMN. In both cases, they use machine learning algorithms and user's previous search history (user profile that contains user's previous queries) for training and testing of the machine. The authors used WEKA for machine learning model creation and AOL dataset. The AOL dataset is spread over three months period in which queries of the first two months were used as training data and last month's queries are used as testing data. The details of their adverse model for both TMN and Tor are discussed as follow.

In the case of TMN, the aim of Peddinti and Saxena's Model was to investigate whether it is possible (and to what extent) for an adverse WSE to filter out TMN queries. For this purpose, they conducted experiments with 60 users selected from AOL dataset. They use PlanetLAB [107], a globally distributed research network to simulate TMN over 60 virtual machines. For fake query generation, they initially use news RSS feeds and then use search results to generate further fake queries. Each virtual machine is configured to maintain its user profile. These user profiles are then used as testing data for the machine learning model. For machine learning model, they consider both supervised and unsupervised machine learning algorithms with defaults parameter settings. However, due to the poor performance of unsupervised algorithms (they employed K-Means and EMCluster) with default parameters, they defer that task for future work. In supervised machine learning algorithms, they select Logistic Regression, Alternating Decision Tree (ADT), Random Forest, Random Tree, and ZeroR. The results showed that their proposed adverse model was able to recognize user queries with an average true positive rate of 48.88%.

In the case of Tor, the aim of Peddinti and Saxena's Model was to associate queries coming out of Tor nodes to the users who issued these queries. For this purpose, they again conducted experiments with 60 users selected from AOL dataset. They assumed that WSE has the list of potential Tor users who are identifiable based on their search query pattern and cookies [108]. They generated Tor-based simulated WSE logs using different settings such as N=100 and 1000, where N shows the number of Tor nodes between user and WSE. For machine learning model they again initially consider unsupervised algorithms, however then shifted their work to supervised algorithms due to the poor performance of unsupervised algorithms. Based on the textual nature of the queries data, Support Vector Machine (SVM) is used as a classification algorithm. The results showed that their proposed adverse model was able to recognize user queries with an average true positive rate of 25.95% for N=100 and 18.95% for N=1000.

2.2.3.2 Balsa's Model

Balsa et al. [109] proposed an abstract model and analysis framework to evaluate the privacy provided by six major profile obfuscation techniques: TMN [35], GooPIR [36], PDS [76], PRAW [75], OQF-PIR [66], and NISPP [110]. Their analysis framework uses DCA (Dummy Classification Algorithm: an algorithm used to detect fake queries), PFA (Profile Filtering Algorithm), and SCA (Semantic Classification Algorithm) to evaluate the selected profile obfuscation schemes. The aim of their analysis framework was to identify the crucial elements that must be considered for security analysis and to evaluate the effectiveness of a fake query generation process. They reported that TMN, GooPIR, PDS, and NISPP are vulnerable to DCA based attacks. Similarly, TMN, PDS, PRAW, OQF-PIR, and NISPP are found vulnerable to PFA (Profile Filtering Algorithm) based attacks.

2.2.3.3 Gervais's Model

Gervais et al. [111] proposed the Linkage Attack to quantify the level of privacy provided by the popular indistinguishability solution i.e., TMN. They proposed a generic model to determine the relation between user and queries using the user's web search behavior. Their proposed model uses a similarity score to identify the source of the query. To model the user search behavior and to find a link between user and queries, they used 5 parameters. These parameters are user identity u, query q, query time t, reply of WSE r, and set of pages clicked by the user c. For example if e is an event that is composed of all previously mentioned information (as shown in Equation 2.16), then the linkage function is used to calculate the similarity between two events. As shown in Equation 2.17.

$$e = (u, q, t, r, c) \tag{2.16}$$

$$LinkageFunction(e_i, e_j) \tag{2.17}$$

2.2.3.4 SimAttack

Petit et al. proposed an adverse model based on their proposed SimAttack [2, 46] to evaluate the level of protection offered by three well-known unlinkability and indistinguishability solutions for private web search i.e., TMN, Tor, and GooPIR. Furthermore, they also evaluate combined indistinguishability over unlinkability solution using SimAttack. They used the vector space model (VSM) to compute the similarity between the user profile (web search history) and query. In SimAttack, each query modeled as a binary vector (Equation 2.18):

$$q = (v_{t1}, v_{t2}, v_{t3}, ...v_{tk}) \tag{2.18}$$

where $v_{t1}, v_{t2}, v_{t3}, ...v_{tk}$ are a binary value that shows the presence of different terms $t_1, t_2, t_3, t_4.....t_k$ in a query q. Similarly, the user profile Pu (Equation 2.19) is a set of non-protected queries sent by the user u before using the protection mechanism.

$$Pu = (q_1, q_2, q_3, ...q_j) \tag{2.19}$$

For a similarity calculation between user profile Pu and query $q(sim(Pu, q))$, they considered the Jaccard index (Equation 2.20 [112], Cosine similarity (Equation

2.21 [113], and Dice's coefficient (Equation 2.22 [114]. However, Dice's coefficient gave maximum *precision* and *recall* rate, therefore they used the Dice's coefficient as similarity calculation metric. All three similarity metrics are expressed as follows:

$$Jaccard(a,b) = \frac{a \cdot b}{\left|\left|a\right|\right|^2 + \left|\left|b\right|\right|^2 - a \cdot b} \tag{2.20}$$

$$Cosine(a,b) = \frac{a \cdot b}{\left|\left|a\right|\right| \cdot \left|\left|b\right|\right|} \tag{2.21}$$

$$Dice'sCoefficient(a,b) = 2 \cdot \frac{a \cdot b}{\left|\left|a\right|\right|^2 + \left|\left|b\right|\right|^2} \tag{2.22}$$

In the case of unlinkability (Tor), the aim of the SimAttack was to identify the requestor of a specific query or query of interest (QoI). For this purpose, Petit et al. used a similarity metric to calculate the similarity between the query and the profiles of the suspected user list. The profile with maximum similarity was considered as query requestor. Petit et al. conducted unlinkability attack experiments with three subsets of AOL datasets containing 100, 200, and 300 users. They evaluated the unlinkability solutions from a different angle such as the impact of the number of users in the system, the number of the user profile, size of the user profile, etc. In terms of privacy, SimAttack was successfully able to identify the original requestor of 36.8% queries with 41.4% *precision*.

In the case of indistinguishability (TMN and GooPIR), the aim of SimAttack was to differentiate between the user query and the fake query. For this purpose, Petit et al. again used Dice's coefficient based similarity matrix to compute the similarity between the user profile and set of queries that contained both fake and original queries. For a fake query generation in case of TMN, they used RSS feed while in case of GooPIR, they used dictionary created from queries of AOL dataset. For experimentation, they used a subset of AOL dataset with 100 users. In the case of TMN, SimAttack was successfully able to classify 46.9% real and

Table 2.5: Comparison of Adverse Models for Private Web Search.

Adverse Model	Target Private Web Search Technique			Algorithms
	Unlinkability	Indistinguishability	PIR Protocols	
Peddinti and Saxena [48]	Tor	TMN	✗	Logistic Regression, Alternating Decision Tree, Random Forest, Random Tree, Zero R
Balsa et al. [109]	✗	TMN, PDS, PRAW, NISSP OQF-PIR,	✗	Dummy Classification Algorithm, Semantic Classification Algorithm Profile Filtering Algorithm
Gervais et al. [111]	✗	TMN	✗	Linkage Function
Petit et al. [2]	Tor	TMN, GooPIR	✗	SimAttack

✗ stands for not designed for this Technique.

fake queries with the *precision* of 87.1%. In the case of GooPIR, SimAttack was able to identify more than 50% fake queries.

Apart from indistinguishability and unlinkability solutions, Petit et al. also conducted some experiments to evaluate the performance of their SimAttack in case of indistinguishability over unlinkability protection mechanism. For this purpose, they simulated the TMN over the unlinkability solution (Tor) to protect the get the anonymized log. Next, the authors combined both the SimAttacks proposed for indistinguishability and unlinkability solutions. They reported that SimAttack was able to identify 35.4% real queries to the correct user with the *precision* of 14.7%.

Table 2.5 shows the summary and comparison of Adverse Models presented in Section 2.2.3 with their target Private Web Search Technique.

2.3 Private Web Search Metrics

Private web search techniques are based on a large number of methods (e.g., proxy service, query modification, query exchange, etc.). This variety of techniques prompted the researchers to evaluate their work differently. This section introduces the metrics used to evaluate user privacy (protection). Moreover, this section also introduces the metrics used to evaluate the performance of privacy attacks and theoretical evaluation approaches.

2.3.1 Privacy Quantification Metrics

The privacy quantification methods are used to quantify the knowledge, which the adverse search engine has obtained with and without privacy protection mechanisms. Thus these methods calculate the privacy exposure. Several methods have been proposed in the literature:

2.3.1.1 Entropy

Entropy is a well-known measure to estimate the quantity of information represented by random discrete variable [115]. In terms of privacy quantification, entropy measures the amount of information an adversary has about a user of interest. The user of interest is modeled by a discrete random variable. For instance, X is the user of interest and the set of keywords used by the user $\{x_1, x_2, x_3, ...x_n\}$ is the sample space, then the entropy of $H(X)$ is expressed as follows (Equation 2.23):

$$H(X) = E\big[-log_2\big(p(X)\big)\big] = -\sum_{i=1}^{n} p(x_i) \cdot log_2\, p(x_i) \tag{2.23}$$

where, p shows the probability mass function of the user of interest (UoI) or discrete random variable X. In other words, the probability that a keyword x_i was used by the user of interest (UoI).

2.3.1.2 Degree of Anonymity

The entropy is also used by Diaz et al. to calculate the degree of anonymity provided by unlinkability techniques [116]. The degree of anonymity is used to calculate the ability of an adverse web search engine to associate the query back to its original requester. Let X represents the requester (i.e. discrete random variable), the entropy of X is shown in Equation 2.23 where $p(x_i)$ shows the probability that the user x_i is the original requester of the query. Let the maximum entropy of the system is represented by H_M (Equation 2.24) and n represents the total number of users in the system. The degree of anonymity D_A is shown in Equation 2.25.

$$H_M = log_2(n) \tag{2.24}$$

$$D_A = 1 - \frac{H_M - H(X)}{H_M} = \frac{H(X)}{H_M} \tag{2.25}$$

In other words, the degree of anonymity shows the quantity of information learned by the adverse web search engine using user's queries. The value of the degree of anonymity varies between 0 and 1 (zero shows that one user is identified as requestor with the probability of 1 (i.e., minimum privacy)). While 1 shows that all users are potential requestor (i.e., maximum privacy) with the probability of $\frac{1}{n}$.

2.3.1.3 Search Engines Results Assessment

Another method used to quantify privacy is to assess the quality of results evolved with the introduction of fake queries. Unlike other web search engines, Yahoo! provides the lists of interests infers from user queries. TrackMeNot (TMN) [34] uses these interests lists to study the effectiveness of its mechanism. They quantify the evolution in the user's interests after using TrackMeNot (TMN) using the lists published by well-known web search engine Yahoo!.

2.3.1.4 Profile Exposure Level

As the entropy is used to calculate the knowledge about the users that an adverse web search engine obtained. However, this knowledge needs to be compared with the knowledge obtained without any privacy preserving technique. For this purpose, Viejo et al. propose Profile Exposure Level (PEL) [56, 65] metric which is used to measure the percentage of exposed information. Let we consider two random variables X and Y where X represents the queries sent by the user without any protection technique and Y represents the queries sent through protection technique. The PEL is defined as follow (Equation 2.26):

$$PEL = \frac{I(X,Y)}{H(X)} \cdot 100 \tag{2.26}$$

where, $H(X)$ shows the entropy of the original set while $I(X,Y)$ shows the mutual information between variable X and Y.

2.3.1.5 Kullback-Leibler (KL) Divergence

Kullback-Leibler (KL) Divergence is used to measure the difference between one probability distribution with other reference probability distributions [117]. It is also known as relative entropy and Rebollo-Monedero and Jordi Forné used this measure to find the divergence between prior and posterior distribution [118]. In other words, they used KL Divergence to find divergence between to profile to assess the privacy leakage. Let we consider two probability distributions P and Q on some probability space X, the KL divergence of Q from P is shown in Equation 2.27:

$$D_{KL}(P||Q) = \sum_{x \in X} P(x) \log \left(\frac{P(x)}{Q(x)} \right) \tag{2.27}$$

Similarly for calculating average divergence between two vectors (User profile and query), the formula for Kullback-Leibler Divergence is shown in Equation 2.28:

$$D_{AvgKL}(P,Q) = \sum_{j=1}^{n} \left(\alpha_P \times \left(W_jP \times ln\left(\frac{W_jP}{W_j}\right)\right) + \alpha_Q \times \left(W_jQ \times ln\left(\frac{W_jQ}{W_j}\right)\right)\right) \tag{2.28}$$

where for j^{th} term
$\alpha_P = \frac{W_jP}{W_jP+W_jQ}, \alpha_Q = \frac{W_jQ}{W_jP+W_jQ}\; and\, W_j = \alpha_p \times +W_jP + \alpha_{Qv} \times W_jQ$

and j represents the components of the vector.

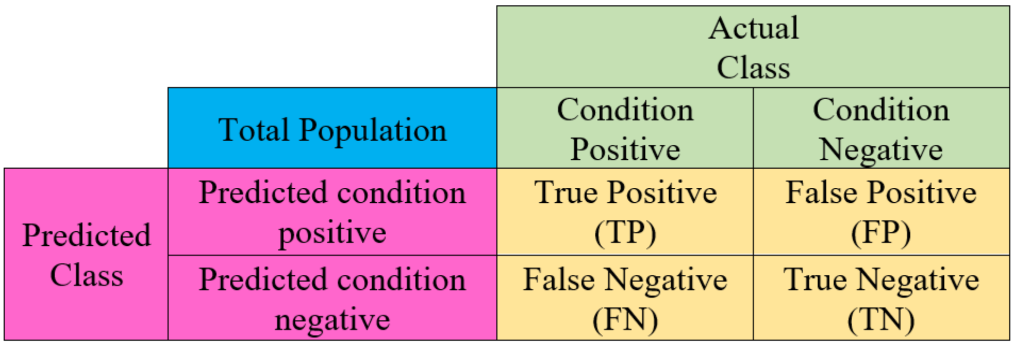
2.3.1.6 Cross-Entropy Loss

Cross-Entropy Loss is used to measure the difference between two probability distributions. In the case of privacy exposure, Cross-Entropy Loss can be used to determine the difference or loss between the user's actual profile and profile created through any privacy protection mechanism [119]. Let us consider P as the user's actual profile and Q is the profile created through any privacy protection mechanism. Then the Cross-Entropy Loss between P and Q is shown in Equation 2.29:

$$H(P,Q) = -\sum_{i=1}^{n}(P_i) \cdot log\,(Q_i) \tag{2.29}$$

2.3.2 Metrics used in Machine-Learning Based Privacy Attacks

Machine-learning attacks are based on supervised and unsupervised machine learning algorithms. The aim of the attack varies with respect to the privacy-preserving solutions. In some cases, the aim of the attack is to identify the originator of the anonymous query, while in some cases the attack is used to identify the fake and real queries. In the case of supervised learning [45, 48], the machine is trained with prior knowledge. The prior knowledge is the user search history (in the case of anonymous queries) and set of fake queries (in case of indistinguishability techniques). Similarly, in unsupervised learning [48, 95, 111], no prior knowledge is



		Actual Class	
	Total Population	Condition Positive	Condition Negative
Predicted Class	Predicted condition positive	True Positive (TP)	False Positive (FP)
	Predicted condition negative	False Negative (FN)	True Negative (TN)

FIGURE 2.2: Confusion Matrix

required. The unsupervised learning algorithm splits queries into two groups: one for real queries and one for fake queries.

The performance of these attacks is usually evaluated in terms of *precision* and *recall*. However, in some cases, the *F-Measure* (harmonic means of *precision* and *recall*), *Accuracy*, and *True positive* rate are also used for evaluation. The values of *precision* and *recall* are using confusion matrix of error matrix [120]. The confusion matrix is a special type of contingency table use to visualize the performance of machine learning algorithms (as shown in Figure 2.2).

$$Precision = \frac{TP}{TP + FP} \tag{2.30}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.31}$$

The true positive (TP) shows the number of instances correctly predicted as positive by the machine, while false positive (FP) shows the number of instances incorrectly predicted as positive by the machine. Similarly, false negative (FN) shows the number of instances incorrectly predicted as negative by the machine while true negative (TN) shows the number of instances correctly predicted as negative by the machine [121]. The terminologies derived from the Confusion Matrix are (*Precision* (Equation 2.30), *Recall* (Equation 2.31), *F-Measure* (Equation 2.32) and *Accuracy* (Equation 2.33).

$$F \cdot Measure = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.32}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.33}$$

2.3.3 Theoretical Analysis

Apart from statistical analysis, some private Information retrieval solutions also use theoretical properties (such as unlinkability between users and their queries) to evaluate the performance of their proposed solutions in terms of privacy. These properties can be verified theoretically using different protocol verifiers such as Avispa [122] and ProVerif [123]. In these verifiers, the protocols are modeled using domain-specific languages Pi Calculus (π-calculus) and testing data is supplied to the developed model. The verifier model derives all possible states in the protocols. The major drawback of these verifiers is the modeling of protocols since the small difference might hide the potential flaw. Well known protocol verification tools are Avispa [122] and ProVerif [123]

2.4 Query Categorization Techniques

Query categorization or classification are used for various purposes in a private web search domain. For instance, sensitive query detection [2], interest extraction [28, 72, 77, 80, 124], fake query generation [34, 44, 79, 109, 110], de-anonymization [14, 46, 72, 111] etc. Initially, query classification was studied and used for the improving quality of the web search engine's results. Many web search services providers use different query categorizers to remove irrelevant results. Query categorization problem was highlighted in 2005 when SIGKDD (Special Interest Group on Knowledge Discovery and Data Mining) organized a competition (KDDCUP 2005) on *Internet user Search and query categorization*. The aim of the competition was the classification of 800,000 queries into 67 categories. Since then, many query classification techniques have been developed that use different online directories query classification. This section contains discussion about popular online directories that are used for query classification.

2.4.1 Query Categorization using Open Directory Project

Open Directory Project (also known as Directory Mozilla (DMOZ[5])) is the largest human-edited index of the websites in which web pages are categorized under hierarchical ontology of topics. Against each query, ODP return most relevant web pages and then ontology is used to map the list of categories. In the end, categories are ranked according to the frequency and the major and sub-categories are returned to the user.

2.4.2 Query Categorization using Wikipedia

Another approach for query classification is based on Wikipedia. This technique was proposed by Alemzadeh et al. [125]. In this technique, the Wikipedia[6] page is extracted against each word of the query. Then using density function, all categories associated with the Wikipedia pages are extracted.

2.4.3 Query Categorization using uClassify

uClassify[7] is another online service that provides different classifiers for age, topics, language, gender detection, and many others. The Topics classifier of uClassify service offer the numeric values of 10 categories against each topic. The topic classifier uses a subset of topics from the Open Directory Project (ODP) directory in which topics are placed in a hierarchy. The classes are Arts, Business, Computers, Games, Health, Home, Recreation, Science, Society, and Sports. The classifier provides the percentage of each query in each category.

[5]http://dmoz-odp.org
[6]https://www.wikipedia.org
[7]https://www.uclassify.com

Chapter 3

QuPiD Attack for Private Web Search

3.1 Introduction

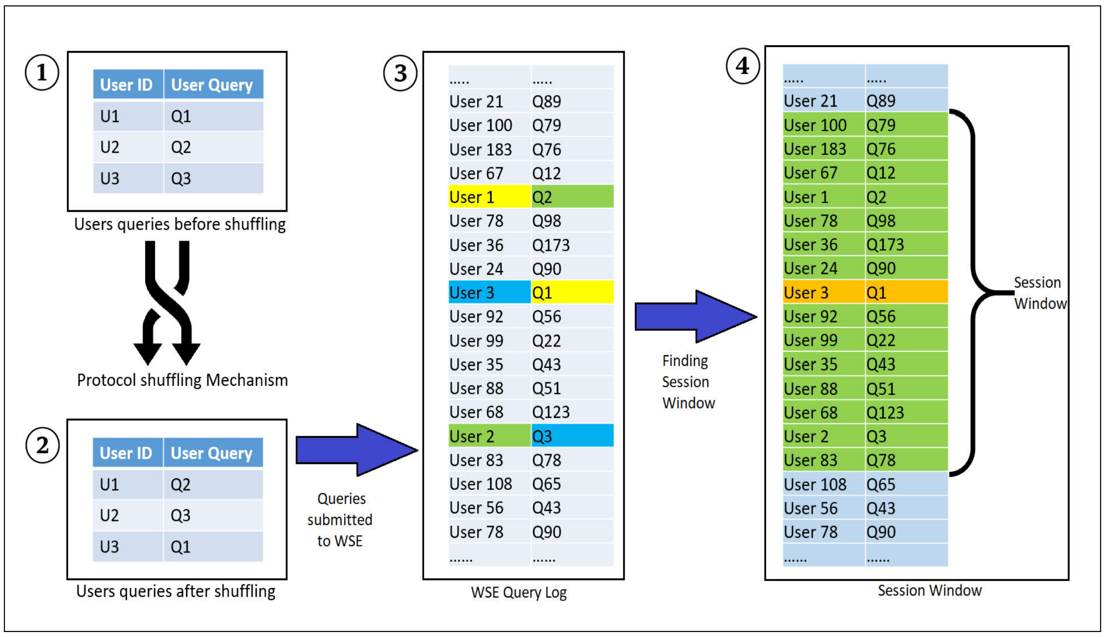
This chapter presents the QuPiD attack, an attack to evaluate the prominent Private Information Retrieval (PIR) protocols i.e., User Private Information Retrieval (UPIR) [41, 42, 68] and Useless User Profile (UUP) [37, 38, 40]. Both UUP and UPIR shuffle protocols the queries in user groups in order to hide the identity of the user. In both protocols, the web search engine receives the user query, however with a different identity and thus, Web Search Engine cannot identify the originator of the queries. We set out to investigate whether it is possible (and if so to what extent) for an adverse Web Search Engine equipped with users' web search profiles (histories), to link queries coming from the UUP exit-user to the original users and thus, undermine the privacy provided by Private Information Retrieval protocols. We propose an approach that targets Private Information Retrieval protocols. To the best of our knowledge, no existing approach has been published that evaluates the performance of PIR protocols using a machine learning approach. Therefore, we designed a machine learning based attack (QuPiD Attack) to investigate the robustness of PIR protocols for private web search.

In this chapter, we have discussed the design of QuPiD Attack in detail along with the detail discussion about experimental setup including datasets, data pre-processing, selected machine learning algorithms, and evaluation method and metrics. The results of experimental evaluations from different angles are discussed in the last section of this chapter.

3.2 Design of QuPiD Attack

This section presents a detail design discussion about QuPiD attack, an attack against prominent private web search solutions, i.e., User Private Information Retrieval (UPIR) [41, 42, 68] and Useless User Profile (UUP) [37, 38, 40]. In PIR protocols, a group of users exchanges queries with each other in such a way that the identity of the query originator node remains hidden from other group mates. In the next step, all group members submit the received queries to the WSE and results are broadcasted in the group. On the WSE side, the user's query is received in plain text but with a different identity and thus, WSE cannot identify the originator of the queries. We set out to investigate whether it is possible (and to what extent) for an adverse WSE equipped with users' web search profiles (histories) to link the queries coming from the UUP exit-user to the original users and thus, undermine the privacy provided by UUP. The QuPiD Attack determines the distance between the user's Profile (web search history) and upcoming query using a novel Feature Vector and machine learning algorithm. The attack uses the user's web search history to train the classification model (created using a supervised machine learning algorithm). The classification model is then used to associate the upcoming queries to their relevant users and thus de-anonymizes the identity of the actual query originator.

As mentioned earlier, WSE received user queries with different identity due to the shuffling process. Therefore, the entries of the queries will never appear with their true originator in the weblog. However, the weakness of this protocol is the timing of query submission by all group members. After query shuffling step, every group



User ID	User Query
U1	Q1
U2	Q2
U3	Q3

User ID	User Query
U1	Q2
U2	Q3
U3	Q1

.....	.....
User 21	Q89
User 100	Q79
User 183	Q76
User 67	Q12
User 1	Q2
User 78	Q98
User 36	Q173
User 24	Q90
User 3	Q1
User 92	Q56
User 99	Q22
User 35	Q43
User 88	Q51
User 68	Q123
User 2	Q3
User 83	Q78
User 108	Q65
User 56	Q43
User 78	Q90
......	......

.....	.....
User 21	Q89
User 100	Q79
User 183	Q76
User 67	Q12
User 1	Q2
User 78	Q98
User 36	Q173
User 24	Q90
User 3	Q1
User 92	Q56
User 99	Q22
User 35	Q43
User 88	Q51
User 68	Q123
User 2	Q3
User 83	Q78
User 108	Q65
User 56	Q43
User 78	Q90
......	......

FIGURE 3.1: Queries Entry in Weblog and Session window.

member submits the received query to WSE almost at the same time. Due to which their entries appeared close to each other in the weblog. Fig. 3.1 illustrates an example of query entries in the weblog. Figure 3.1, exhibit 1 shows the users' queries before shuffling process while exhibit 2 shows the queries after the shuffling process. After shuffling, the queries are submitted to WSE (exhibit 3).

In the proposed adverse model, WSE is assumed to be an entity whose goal is to work against the privacy-preserving solution and to identify the user of interest (UoI) queries for profiling purposes. It is assumed that WSE is equipped with the user's search history (i.e., user profile) PU. The user profile contains queries submitted by the user in the past without using any UUP protocol as shown in Equation 3.1 (where P_qi shows the queries in the UoI profile).

$$PU = \{P_q1, P_q2, P_q3, ...P_qn\} \tag{3.1}$$

The user profile PU is used as training data for building the classification model. As the dataset used for experimentation is spread across three months duration, the first two months data is used as a training set. While the UUP protocol is simulated using the third-month data to create an anonymized log (as shown in

Figure 3.1, exhibit 3). The anonymized log is used as the test set. For testing, all session windows of the UoI are drawn out from the query logs. The session window is a block of records (queries entries in the log) in an anonymized log that contains the entry of UoI, but with another user [72, 124]. In other words, the session window is composed on the selected number of queries' entries in the WSE query log, appeared immediately before and after the query of UoI. As shown in Figure 3.1 (exhibit 4), UoI is "User 3" and the session window size is 15 records (7 records before UoI and 7 after UoI). In this thesis, we have used different kinds and size of windows depending on the need for the experiments. The details of different types of session windows are given in Section 3.3. Let the total size of the window be $n + 1$ then each session window S_{win} is composed of $\frac{n}{2}$ queries appearing before and $\frac{n}{2}$ queries after the query of UoI. A generic session window S_{win} is shown in Equation 3.2 (where qi represents a query in the session window). The collection of all session windows GS_{win} is shown in Equation 3.3.

$$S_{win} = \{q_1, q_2, q_3, ...q_{\frac{n}{2}}, q_{UoI}, q_{\frac{n}{2}+1}, ...q_n\} \tag{3.2}$$

$$GS_{win} = \{S_{win\,1}, S_{win\,2}, S_{win\,3}, ...S_{win\,n}\} \tag{3.3}$$

As in the query log, the target user who uses any PIR protocol will remain hidden since his/her query is exchanged with a query of another user in the group. Therefore, a session window is used to reduce the testing data. Both PU (training set) and $GSwin$ (testing set) are used as input to the algorithm of the adverse model. The working of the adverse model is presented in Algorithm 1 and depicted in Fig. 3.2. The working of the algorithm is as follow:

1. In the first step, the user profile PU Feature Vector is acquired for training purposes. The user profile with the Feature Vector PUv is shown in Equation 3.4. The Feature Vector is acquired from the uClassify service, a machine learning web service that provides numerous different classifiers for

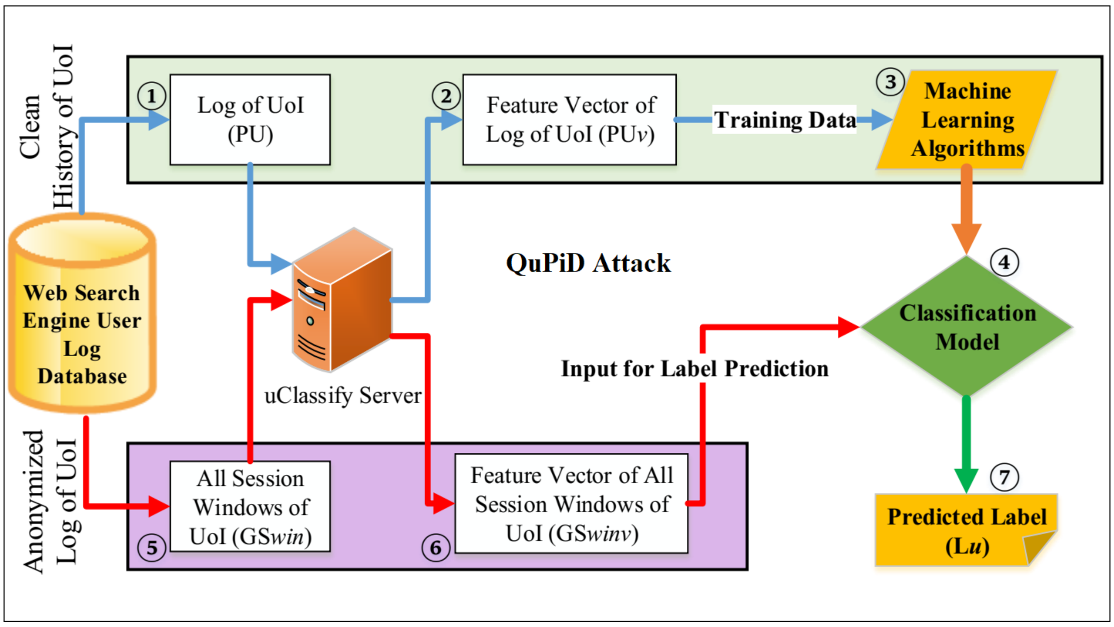

Clean History of UoI
Web Search Engine User Log Database
Anonymized Log of UoI
1 Log of UoI (PU)
2 Feature Vector of Log of UoI (PUv)
Training Data
3 Machine Learning Algorithms
QuPiD Attack
uClassify Server
4 Classification Model
Input for Label Prediction
5 All Session Windows of UoI (GSwin)
6 Feature Vector of All Session Windows of UoI (GSwinv)
7 Predicted Label (Lu)

FIGURE 3.2: Operation of the Adverse Model.

Algorithm 1 Associating incoming query to the user using prior profile.

Input: User Profile (PU), All Session Windows belongs to user (GS_{win}).
Output: Expected User Label (Lu)

```
1: procedure QUERY ASSOCIATION($PU$, $GS_{win}$)
2:     for $P_q i \in PU$ do
3:         $PUv \leftarrow$ get Feature Vector for ($P_q i$)
4:     $P_{Model} \leftarrow$ Classification Algorithm ($PUv$)
5:     for $S_{wini} \in GS_{win}$ do
6:         for $q_k \in S_{winj}$ do
7:             $q_k v \leftarrow$ get Feature Vector for ($q_k$)
8:             $Lu \leftarrow P_{Model}(q_k v)$
9:     return $Lu$
```

text classification. We have selected the "Topics" classifier that gives the score of each phrase or query in 10 major classes.

2. In the second step, a classification model P_{Model} is built using PUv and supervised machine learning algorithms. To test the response of the data with different classification techniques, various machine learning algorithms are used. The details of these algorithms are given in Section 3.4.6.

3. After the classification model P_{Model}, the third step is to acquire the Feature Vector $S_{win\,v}$ (shown in Equation 3.5) for the queries of session window S_{win} from uClassify for testing data.

4. In the last step, each query of $S_{win\,v}$ is provided to the classification model for the expected label Lu. The label Lu shows whether the incoming query belongs to UoI or not.

$$PU_v = \{P_q1_v, P_q2_v, P_q3_v, ...P_qn_v\} \tag{3.4}$$

$$S_{win\,v} = \{q1_v, q2_v, q3_v, ...qn_v\} \tag{3.5}$$

3.3 Session Window

The session window is a block of records in which query/queries associated with the target user are present but might be with together with other user ids (depending on the protocol). In other words, the session window is composed of the selected number of entries (or records) in the web search engine query log which appeared immediately before and after the query of UoI or QoI. The session window is used to reduce the number of records to be tested for associating with queries to the correct user. We have proposed and conducted various types of session window techniques such as query of interest (QoI) based [124], the user of interest (UoI) based [72], and time-based. In QoI based method, a session window comprises of a block of the specified number of records, appearing before and after QoI in the log [124], while in UoI based method, a session window is a block of the specified number of records, appearing before and after UoI in the log [72]. It was concluded from previous investigations that the session window size of 300 offers better results in associating queries to the correct user [72, 124]. Both types of session windows are based on a fixed number of records, which is appropriate for limited datasets. However, as an ordinary user takes 2 to 3 minutes interval between two queries, a session window of 300 records might miss the actual user. Therefore, we proposed the time-based session window to cover all the queries of users submitted in a specified time using any PIR protocol.

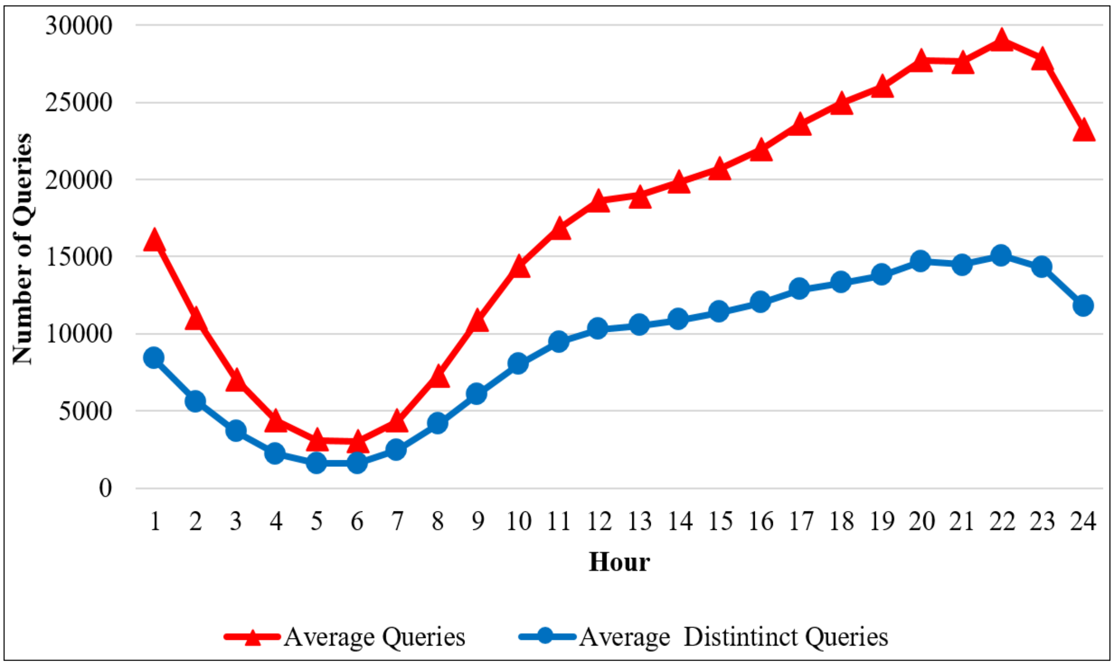


FIGURE 3.3: Traffic and Distinct Query Pattern.

In the time-based session window, a single day is divided into 24 session windows and each window represents an hour of the day. The traffic pattern of a day is shown in Figure 3.3. The pattern is generated from the traffic of randomly selected one week from the AOL dataset. Figure 3.3 contains two patterns: one represents the average queries submitted in a corresponding hour while the other represents average distinct queries submitted during the respective hour.

3.4 Experimental Setup

This section provides a detailed description of the experimental setup, dataset, feature selection procedure, and classification algorithms selection process. The experiments include among others performance evaluation of classification algorithms, significance of Feature Vectors, the impact of group size, the impact of the number of queries, the impact of profile size and the impact of the session window size. The performance of the attack is assessed in terms of *precision*, *recall*, *f-measure*, and other metrics. All experiments were conducted on a workstation with a 4.1 GHz Core i7 processor and 16 GB RAM.

3.4.1 Dataset

This section contains the detail discussion about the dataset and its subsets we used in this research for evaluation of the proposed attack. We used AOL web search log for conducting the experiments. The AOL web search logs consist of queries submitted by over 6.5 million users during the three months period from March 2006 to May 2006. All the queries in the log belong to real users, however, their identity is replaced with the fictitious number and referred as anonymous ID. The dataset is consists of five attributes user ID, query, date and time of the query, clicked content rank and clicked URL. All queries in the dataset include five attributes:

1. **AnonID**: A fictitious number which was assigned to each user instead of their real identity (to ensure anonymity of the users) and represent each user uniquely.
2. **Query String**: The query string which is issued by the user.
3. **QueryTime**: The Date and time of the query which represents at which time the query issued.
4. **ItemRank**: The rank of the item over which user clicked from result page against the requested query. Most of the records don't have this field.
5. **ClickURL**: The URL of the page which was selected by the user in the result page. Most of the records don't have this field.

We only used three attributes for experiments including the AnonID, Query and Query Time of the AOL dataset. For each query, we also obtained the Feature Vector from the uClassify[1] service. uClassify is an online service that provides different kinds of text classifiers such as sentiment analysis, topics, language detection, news classification, spam classification, gender analyzer and many others. The details of the preprocessing steps, Feature Vector, user selection, and subset creation are discussed as follow.

[1]www.uclassify.com

3.4.2 Data Preprocessing

Real-world data is usually in raw form and cannot be exploitable for experimentation. We used various preprocessing steps to transform the data into the exploitable form. Initially, we removed irrelevant data from the query string. Most of the queries were composed on meaningless data such as "dfdf", "-", "asd", empty spaces etc. After removing meaningless queries, we used "stop words removal filter" for further cleaning. The list of the stop words is given in Appendix A. After cleaning the data from meaningless queries and stop words, we used uClassify to acquire the score of queries in ten major topics for the training of machine learning model.

3.4.3 Feature Vector Extraction

As mentioned previously, the dataset is composed of five features: user ID, submitted query, query date and time, the rank of the clicked content, and URL clicked. In the past, several exercises were made to predict the class or cluster of the upcoming query based on the previous history. Peddinti and Saxena used the attributes available in the query log [14, 45, 48]. Gervais et al. used multiple tools and methods to find the distance between the queries like Jaccard, TF/IDF, Levenshtein, ODP categories, and others [111]. Gervais et al. also used "query date and time" and clicked URL attributes to predict the queries, whereas Petit et al. [2, 46] devised their own mechanism SimAttack which uses Dice's coefficient [114]. In most of the works, string similarity mechanisms are prevailing for building the classification models. In this research, however, three major attributes: user ID, submitted query, query date and time are used along with ten acquired attributes that are used to classify the user query into 10 major topics. These ten attributes are acquired from uClassify online service for every query in the dataset. uClassify provides multiple kinds of services for text classification like language detection, gender, topics, age, sentiments, and other criteria. For this research, Topics[2] classifier is used which classifies the textual data into ten major

[2]https://www.uclassify.com/browse/uclassify/topics

Table 3.1: Percentage of Query "Soft Drink" Acquired from uClassify in 10 Major Topics.

Recreation	Health	Home	Games	Business	Arts	Computers	Science	Sports	Society
0.2206	0.1980	0.1689	0.0938	0.0683	0.0631	0.0560	0.0549	0.0398	0.0363

topics i.e., Health, Recreations, Arts, Home, Business, Society, Games, Sciences, Computers, and Science. The classifiers provide the percentage of given text data in each above-mentioned category. For instance, the percentage for query "soft drink" is shown in Table 3.1. From this place on-wards, we will call the newly acquired Feature Vector as "Topics Score".

In order to get the better model, we build classification with three sets of features i.e. (i) Strings (Queries), (ii) Topic Score, and (iii) Strings + Topic Score. It was found that the model build with users' queries and topic score attributes bears good results. The details of these experiments are given section 3.5.2.

3.4.4 User Selection and Subsets Description

The dataset used for the experiments is the query log released by AOL in August 2006. The dataset is composed of 36389567 queries submitted by 657426 users during the period of three months (i.e., March 2006 to May 2006), while the total number of distinct queries were 10154630 [17]. The attributes of the dataset are user ID, query, query time and date, the rank of the clicked item, and the clicked URL.

In the entire dataset, only 3.29% of users issued more than 300 queries while about 86% of users submitted less than 100 queries. Fig. 3.4 shows the number of queries submitted by the users in the whole dataset. In light of the experiments conducted by Petit et al. [2, 46], only 18,164 were found as active users and suitable for the experiments. According to Petit et al., active users are those users who submitted queries for at least 61 days (two-thirds of the dataset) with a consecutive period of 45 days (half of the dataset period). Therefore, out of 18,164 users, a dataset is

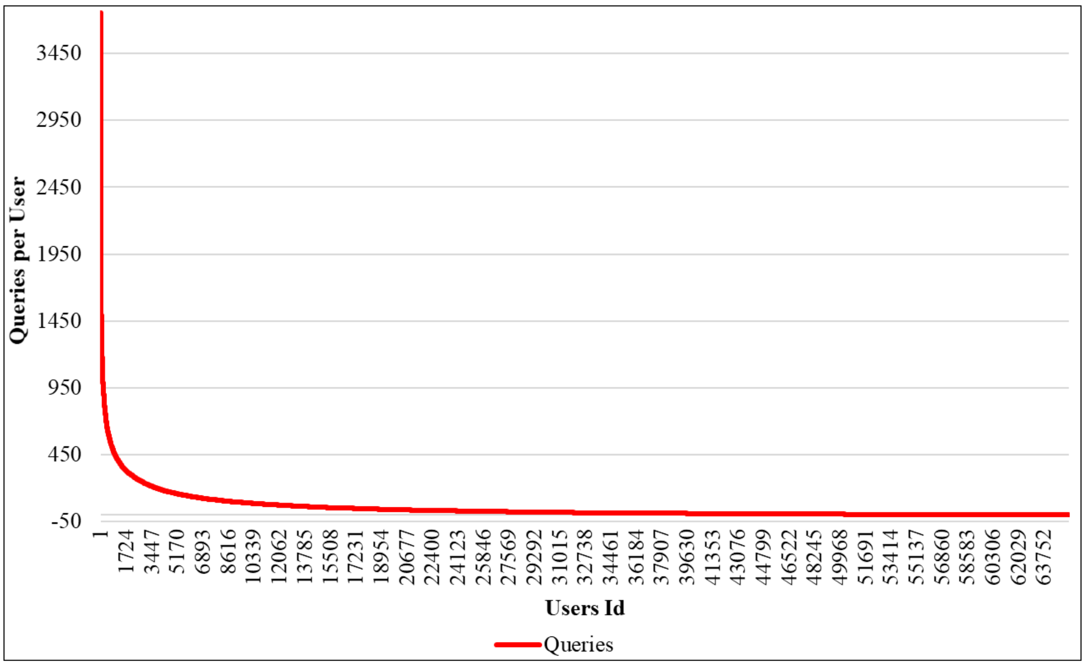


Figure 3.4: Distribution of the Number of Queries issued per User in AOL Dataset.

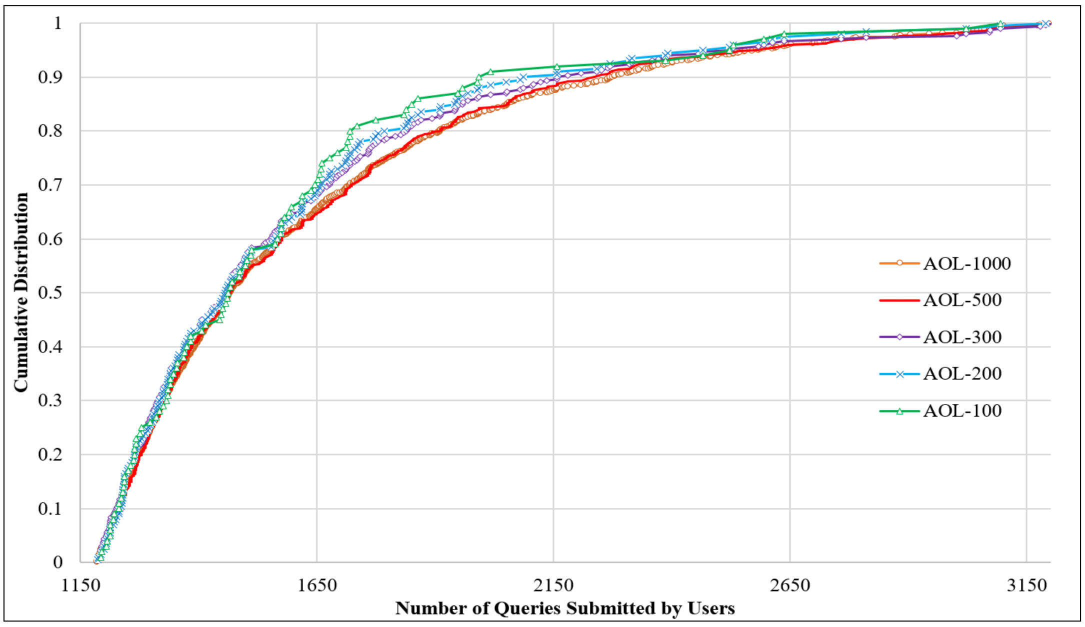


Figure 3.5: Distribution of Number of Queries submitted by per User in Different Datasets.

created that contains top 1000 users with respect to query count, naming as AOL-1000. As the size of AOL-1000 is still huge to conduct experiments, four subsets are created by picking random 100, 200, 300, and 500 users from AOL-1000. The random selection is made to preserve the statistical properties of the data. Fig. 3.5 shows the cumulative distribution of all datasets.

3.4.5 Anonymized Log Creation

As mentioned earlier, the AOL data spans across three months. For experimentation purpose, we have considered the first two months data as the clean history of UoI available to the search engine and last month data as new queries to be classified. We used last month data for the anonymized log. For the creation of the anonymized log, the first step was the protocol parameter setting. The protocol parameters considered in this research are group size, number of queries. As mentioned previously, both the Useless User Profile and User Private Information Retrieval protocols create a group of users, thus the group size is one of the major parameters, although none of the protocols deployed completely due to high computation and communication overheads [14]. However, the Useless User Profile protocol was simulated on a group size of 3, 4, 5, 10 users [37, 38, 54]. Thus for conducting our experiments with regard to the impact of group size and impact of the number of queries, we created logs with different settings. The details of the selected parameters for each experiment are discussed in the description of each experiment in the evaluation section. The code for the log creation is available on Git-Hub[3].

3.4.6 Machine-Learning Algorithms

Machine Learning (ML) algorithm is a very integral part of the proposed model as they are used for building the classification model. In several previous studies, Peddinti et al. [14, 45] and Petit et al. [2] used Random forest, AD Tree, Zero R, Regression, and SVM algorithms for the classification of the data queries. In both studies the classification model was bi-class, i.e., the query is machine or user-generated. Moreover, the model was built based on two attributes like query and assigned label. However, in our work, the classification model is multiclass, i.e., in the testing data the model will decide which query belongs to which user and ML model is based on twelve attributes. Initially we select 10 off-the-shelf

[3]https://github.com/rafiyz/PIR-protocols

(default setting) classification algorithms form different families such as J48 [126] and Logistic Model Tree (LMT) [127] from the tree-based family, Decision Table [128], JRip [129] and OneR [130] from Rule-based family, IBK [131] and KStar [132] from Lazy-Learner family, Bagging [133] and XGBoost [134] from Meta-heuristic family (Ensemble Learning) and Naïve Bayes [135] from Bayes family. Rep Tree [136] and Regression are used as base classifiers for Begging algorithm. However, due to the poor performance of many algorithms, we selected the top five algorithms, i.e. Naïve Bayes, IBk, Bagging, XGBoost, and J48. (The performance comparison of all 10 selected algorithms is available in Appendix B). Parameters of all selected machine learning algorithms are shown in Table 3.2. Moreover, in order to test the performance of the QuPiD attack with advance and sophisticated machine learning algorithms, the QuPiD attack is also tested with Artificial Neural Network (ANN) and results are shown in section 3.5.1.4. A brief introduction of each classifier is provided below:

- **Naïve Bayes:** This algorithm belongs to the probabilistic class of the classifiers which predicts the chance of occurring an event-based analysis of past events [137]. The popular applications of this classifiers are medical diagnostics, spam identification, and text classification [138]. In this research, the most generic version is used with default parameters.
- **IBK:** This algorithm belongs to the distance-based group of the classifiers. It uses k-nearest neighbor classifier to find the distance between two vectors [131]. In this research, the algorithm used to a searching nearest neighbor is ”Linear-NN-Search” which works on ”Euclidean Distance”. The rest of the parameters remain default.
- **Bagging:** Bootstrap Aggregation (Bagging) is meta-heuristic algorithms which use different machine learning algorithms to achieve better prediction [133]. It divides the training data into small datasets and creates classifiers for each dataset based on the selected classifier. The results of all small datasets are then combined using average or majority voting or other methods. For this research, REPTree algorithm is used as base learner.

- **XGBoost:** XGBoost stands for extreme gradient boost is an ensemble machine learning algorithm, used to deal with classification and regression problems. XGBoost is evolved from decision trees algorithm that uses gradient boosting techniques to minimize the error in sequential prediction tree [134]. Extreme Gradient Boost is the evolved version of Gradient Boosting technique that offer parallel processing, tree-pruning, and missing values handling techniques.

- **J.48:** J.48 is tree-based classifier which is based on the C4.5 algorithm [126] (a decision tree based classification algorithm developed by Ross Quinlan). J.48 is a statistical classifier which creates a decision tree-based on information entropy. It is considered as the most prominent and most widely used algorithm for machine learning activities.

- **Artificial Neural Network (ANN):** Artificial Neural Network (ANN) is a computing system inspired by the simplification of neurons in an animal brain [139]. Artificial Neural Network is based on a network of artificial neurons or nodes like a biological brain. Each node receives a signal, processes it, and can send the signal to other neurons connected to it. In ANN signal at connection is a real number and output of each neuron is calculated using some non-linear function of the sum of its input. Usually, neurons are aggregated into layers and each layer may perform different transformations to the input.

  ANN can be classified into six major categories: Radial basis function Neural Network, Feedforward Neural Network, Recurrent Neural Network (RNN), Kohonen Self Organizing Neural Network, Modular Neural Network, and Convolutional Neural Network (CNN). Usually, for problems like identification of picture, CNN is used while for the problems such as sequence to sequence translations (speech or handwriting recognition), RNN is used with Long Short-Term Memory (LSTM). Unlike the simple ANN, in Recurrent Neural Network, a concept of Long Short-Term Memory (LSTM) is used which has the feedback mechanism. This feedback mechanism allows the network to process an entire sequence of the data [140].

Table 3.2: Parameters of Selected Classifiers used in QuPiD Attack.

Classifier	**Parameters**
Naïve Bayes	Batch size = 100 Kernel Estimator = False
IBk	Batch Size =100 KNN value =1 - 7 NN search algorithm = Linear NN Search (Euclidean Distance)
Bagging	Bag Size= 100 Batch Size =100 Iterations = 10 Classifier = REP Tree Seed = 1 Num Folds = 3 Max Depth = -1

XG Boost	Verbosity = 1
	Iteration = 250
	Learning Rate = 0.3
	Seed = 1
	Sub Sample = 0.5

J48	Confidence Factor = 0.25 Seed =1 No of Folds = 3 Batch size = 100 Pruning = False
ANN	Type = Recurrent Neural Network Layer = LSTM Activation Function = Activation ReLU Gate Activation Function = Activation Sigmoid No of Epochs = 5, 10, 15, 20, 25, 30, 50, and 100

Where **Batch Size** shows the number of instances to process in case of batch prediction. **Kernel Estimator** is a weighting function used in non-parametric technique. **KNN** value shows the number of neighbors to used for calculation. **NN Search Algorithm** is the Algorithm used to find the nearest neighbor. **Bag Size** is used to define the size of bag for training data. **Iterations** parameter is used to specify the number of iterations to be performed by the algorithm. **Classifier** parameter specifies the base Classifier used by in bagging classifier. **Seed** parameter is used for randomize the data. **Num folds** specifies the amount of data used for pruning. **Max Depth** shows the maximum tree depth (-1 shows no restriction). **Verbosity** shows whether to print the early stopping information or not. **Learning Rate** shows that how quickly the error is corrected from each tree to the next. **Sub Sample** means that XGBoost would randomly sample half of the training data prior to growing trees to avoid the problem of over-fitting. **Confidence Factor** is used for tree pruning. **No of folds** specifies the amount of data used to reduced-error pruning. **Type** shows the class of Artificial Neural Network used. **Layer** specifies the type of layer used. **Activation Function** of a node defines the output of that node given an input or set of inputs. **Gate Activation Function** defines the activation function to use for the gates. **No of Epochs** is the number of complete passes through the training dataset.

3.4.7 Evaluation Metrics

To measure the efficiency of the adverse model (QuPiD Attack) with different classification algorithms, three well-known metrics i.e., *precision* (Equation 3.6, *recall* (Equation 3.7), and *f-measure* (Equation 3.8) are used. The mathematical representations of all three metrics are as follow:

$$precision(j) = \frac{TP(j)}{TP(j) + FP(j)} \quad (where\ j \in U) \tag{3.6}$$

$$recall(j) = \frac{TP(j)}{Q(j)} \quad (where\ j \in U) \tag{3.7}$$

$$f \cdot measure = 2\,\frac{precision(j) \cdot recall(j)}{precision(j) + recall(j)} \quad (where\ j \in U) \tag{3.8}$$

where U represented the set of UoI (User of Interest) and $Q(j)$ represented the set of queries issued by the user using any PIR protocol. $TP(j)$, the subset of $Q(j)$ represented the set of queries submitted by the user $U(j)$ and successfully retrieved by the Adversarial Model. While $FP(j)$ representing the set of those queries which did not belong to $Q(j)$ but were mistakenly considered as queries of $U(j)$, $f{\cdot}measure$ is a harmonic mean of *precision* and *recall* representing the trade-off between *recall* and *precision.* As the data set was too large and it was difficult to represent *precision*, *recall*, and $f \cdot measure$ of all users, the weighted average of all three (weighted *precision*, *recall*, and *f-measure*) measures were considered. They are mathematically defined as follow:

$$weighted\,average\,precision = \frac{1}{Q}\sum_{j\in U} precision(j)\cdot\Big(TP(j)+FP(j)\Big) \tag{3.9}$$

$$weighted\,average\,recall = \frac{1}{Q}\sum_{j\in U} recall(j)\cdot\Big(TP(j)+FP(j)\Big) \tag{3.10}$$

$$weighted\,average\,f\cdot measure = \frac{1}{Q}\sum_{j\in U} f\cdot measure(j)\cdot\Big(TP(j)+FP(j)\Big) \tag{3.11}$$

where Q represents the total number of queries issued by all Users of Interest (UoI).

3.5 Evaluation

In this section, we evaluate the privacy protection offered by PIR protocols with QuPiD attack. We conducted four major experiments to assess the capabilities of QuPiD attack under various situations. The experiments included the impact of group size, the impact of the number of the queries, the impact of profile size, and impact of session window size. In addition, we conducted two additional experiments, one to evaluate the classifier and Feature Vector for the selection of the best classifier, and one to assess the performance significance of different Feature Vectors. During the investigations, an unexpected behavior of the proposed model was observed in some cases, which predominantly affected the rate of *recall* and *precision*. We call this behavior a ProQSim (Profile to Query Similarity) Effect. The details of ProQSim Effect are also discussed in this chapter.

3.5.1 Classifier Evaluation

This section contains the results and discussions about the various experiments conducted with varying parameters to evaluate the performance of QuPiD attack and the performance of selected protocols in terms of privacy provided from different angles. In pursuit of the best classification algorithm, all five selected

TABLE 3.3: Classifiers Evaluation.

Classifier	**Precision**	**Recall**	**F-Measure**	**Model Build Time (Sec)**	**Testing Time (Sec)**
IBk	0.766	0.451	0.514	0.01	15.12
XGBoost	0.745	0.440	0.488	10.2	4.17
Bagging	0.743	0.430	0.487	4.3	0.2
J48	0.709	0.422	0.477	2.13	0.55
Naïve Bayes	0.743	0.388	0.452	0.72	0.77

classification algorithms were tested on 20 different sample datasets. Each sample dataset consisted of 37000 records with a ratio of 66:34 for Training and Testing. The results of the experiments are shown in Table 3.3, Fig. 3.6, and Fig. 3.7 in terms of *average precision*, *average recall*, model build time, and testing time. Moreover, in order to avoid the problems like over-fitting and selection bias, the results of experiments regarding 10-Fold Cross Validation for each classifier are discussed in section 3.5.1.1.

The results show that Bagging, IBK, and XGBoost give almost similar and maximum *precision* among all selected classifiers, but IBK gives better *recall*. J.48 is another potential candidate with a slightly lower rate of *precision* and *recall* rate. The performance of Naïve Bayes was a complete disappointment both in terms of *precision* and *recall*, although we had hoped for better results as it works on probability. In terms of time, the performance of every algorithm was fabulous. Model building and testing time were almost negligible for every classifier except XGBoost with 10.2 seconds for model building and IBK with 15.12 seconds to test the data.

From the results, it is clear that the performance of IBk is better than the rest of the selected classifiers. However, In order to further investigate the performance of the QuPiD attack with selected classifiers, we also conducted some other experiments as well including 10-fold cross-validation, variable training, and testing ratio, the performance of IBk with different K values and performance of the QuPiD attack with Artificial Neural Network (ANN). The conclusions of experimental evaluations of classifiers are discussed in section 3.5.1.5.

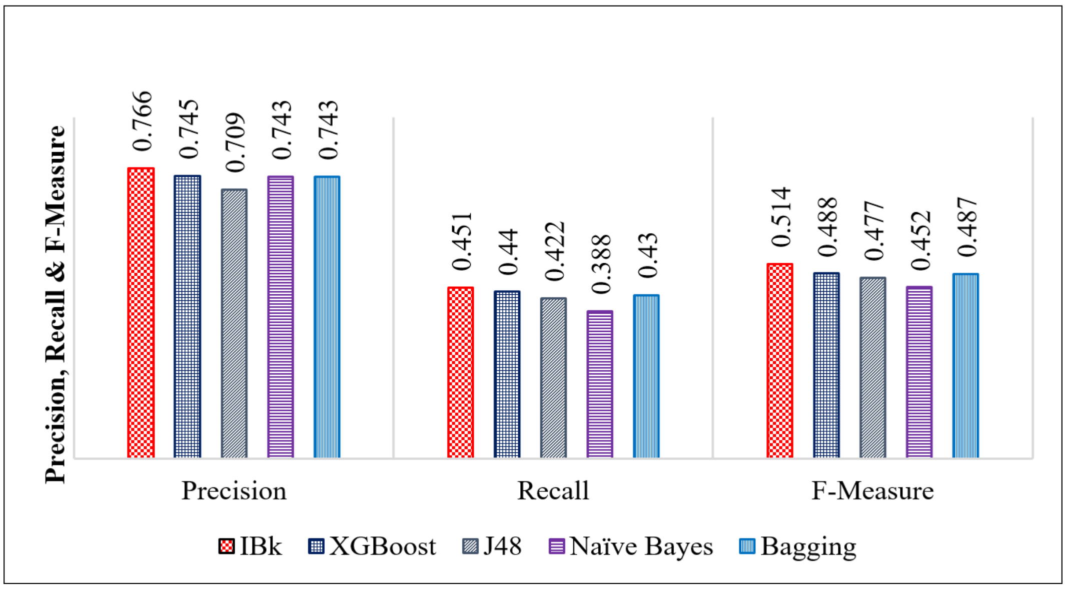


FIGURE 3.6: Classifier Accuracy Evaluation.

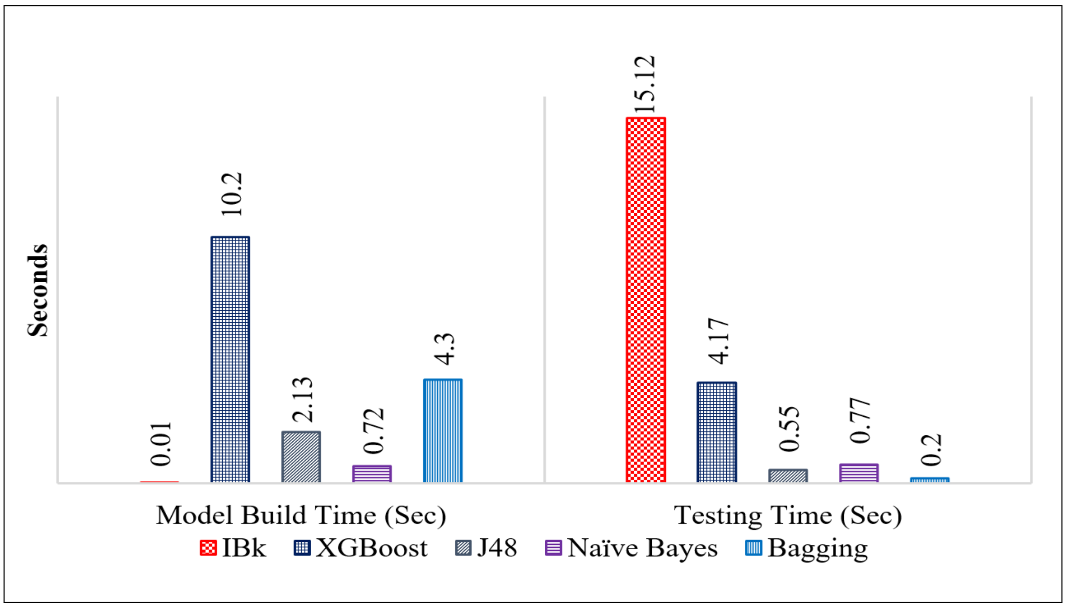


FIGURE 3.7: Classifiers Time Evaluation.

3.5.1.1 Performance of QuPiD Attack under Selected Classifiers with 10-Fold Cross Validation

Cross-validation is a statistical technique used to evaluate the ability of machine learning models. It is usually used for the prediction models to estimate the accuracy of the model in practice. The goal of cross-validation is to test the ability of the perdition model in order to avoid problems such as selection bias and over-fitting [141]. Cross-validation is a re-sampling procedure used to evaluate

TABLE 3.4: Performance of Classifiers with 10-Fold Cross-Validation.

Classifier	Precision	Recall	F-Measure
IBk	0.773	0.462	0.526
XGBoost	0.749	0.444	0.503
Bagging	0.709	0.418	0.471
J48	0.678	0.408	0.459
Naïve Bayes	0.701	0.344	0.48

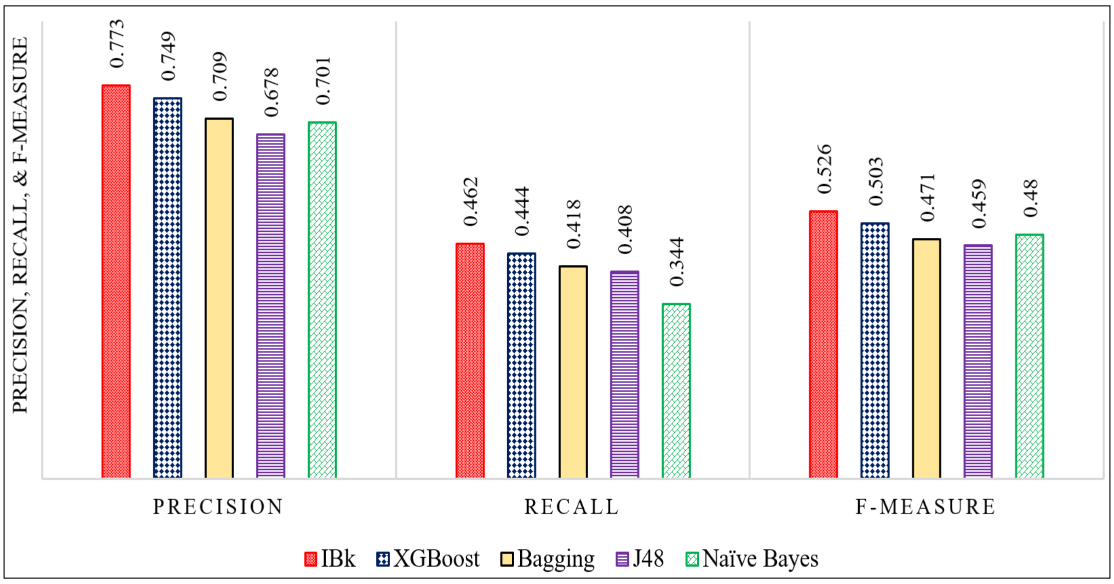


FIGURE 3.8: Performance of Classifiers with 10-Fold Cross-Validation.

machine learning models on a limited data sample. The procedure has a single parameter called ”k” that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called ”k-fold” cross-validation. When a specific value for k is chosen, it may be used in place of ”k” in the reference to the model, such as k=10 becoming 10-fold cross-validation [142]. In this section, we discussed the ability of the selected classifiers with 10-Fold Cross-Validation.

The results show that IBk performed better as compare to the rest of the classification algorithms both in terms of *precision* and *recall*. While the performance of XG Boost and Bagging was comparatively better than the J48 and Naïve Bayes. It can be easily concluded from the results that the QuPiD attack performed well with the IBk classification algorithm. The results of the 10-Fold Cross-Validation of all selected classification algorithms are shown in Table 3.4 and Fig. 3.8.

TABLE 3.5: Performance of Classifiers with Different Training and Testing Ratios.

	Training and Testing Ratio	Bagging	IBk	J 48	Naïve Bayes	XG Boost
Precision	60:40	0.727	0.777	0.722	0.701	0.727
	70:30	0.76	0.77	0.721	0.7	0.76
	80:20	0.779	0.812	0.753	0.702	0.779
	90:10	0.799	0.825	0.772	0.701	0.799
Recall	60:40	0.409	0.44	0.409	0.306	0.419
	70:30	0.417	0.45	0.419	0.309	0.427
	80:20	0.422	0.454	0.422	0.32	0.432
	90:10	0.429	0.456	0.427	0.35	0.439

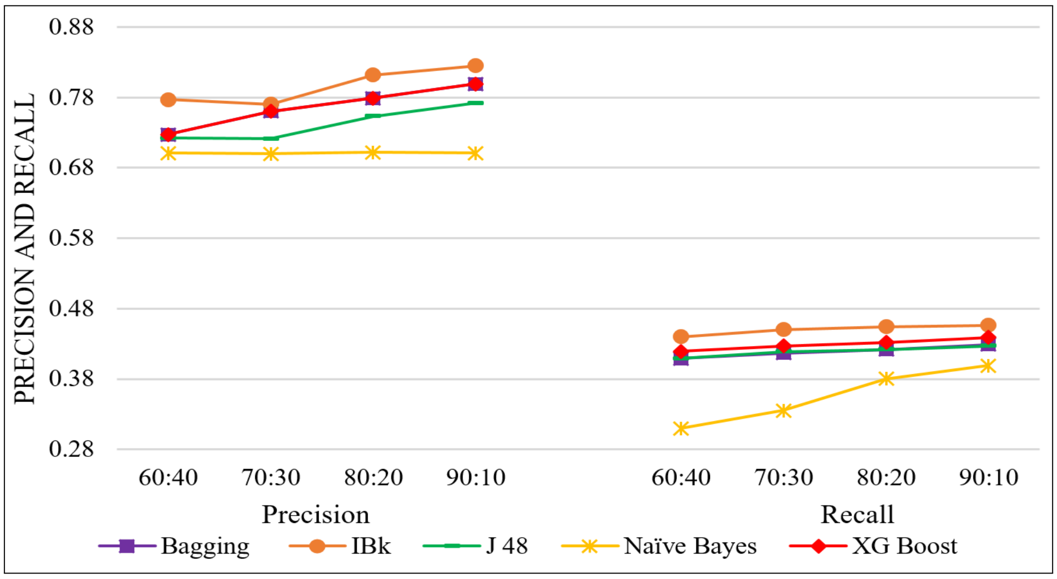


FIGURE 3.9: Performance of Selected Classifiers with Different Training and Testing Ratios.

3.5.1.2 Performance of Selected Classifiers with Different Training and Testing Ratios

It is one of the main requirements in machine learning to build a model with high prediction as possible but generalization capabilities [143]. In supervised learning (classification), a prediction model is built to predict the output of unknown data based on the training examples. However, the success rate of the prediction model is dependent upon the amount of training data. Therefore in order to test the capabilities of the selected classification algorithms, we conducted the experiments with different training and testing ratios including 60:40, 70:30, 80:20, and 90:10. The results of these experiments are shown in Table 3.5 and Fig. 3.9.

TABLE 3.6: Performance of QuPiD Attack under Different K Values of IBk Algorithm

K Value	Precision	Recall	F-Measure
K = 1	0.766	0.451	0.514
K = 2	0.7	0.415	0.469
K = 3	0.656	0.391	0.438
K = 4	0.624	0.373	0.416
K = 5	0.598	0.36	0.399
K = 6	0.578	0.348	0.384
K = 7	0.561	0.339	0.373

The results show that IBk yields better results as compared to other selected classification algorithms both in terms of *precision* and *recall* in all scenarios. However, the rate of *recall* for IBk fluctuates between 0.44 and 0.456 which is almost similar. Therefore it is safe to conclude from the results that IBk bears almost the same result with ratios between 60:40 and 90:10 with slight improvement.

3.5.1.3 Performance of QuPiD Attack under Different K Values of IBk Algorithm

From the experiments reported in previous sections, it was concluded that the performance of the QuPiD attack is better with the IBk classifier. IBk is a distance-based algorithm and uses the K-Nearest Neighbor (KNN) algorithm to find the distance between two vectors. Where "K" is a parameter that refers to the number of nearest neighbors to include in the majority of the voting process. "K" value is used to deal with the effect of noise on the classification process [144]. This section contains the discussion about the performance of the QuPiD attack with K=1 to K=7. The results of these experiments are shown in Table 3.6 and Fig. 3.10.

The results show that the performance of the QuPiD attack is getting worse with the increase in K value. The *precision* and *recall* rate is 0.766 and 0.451 respectively with K=1 while with the increase in K value, the rate of *precision* and *recall* is gradually dropping. Therefore it is concluded from the results that the QuPiD attack gives better results with K=1.

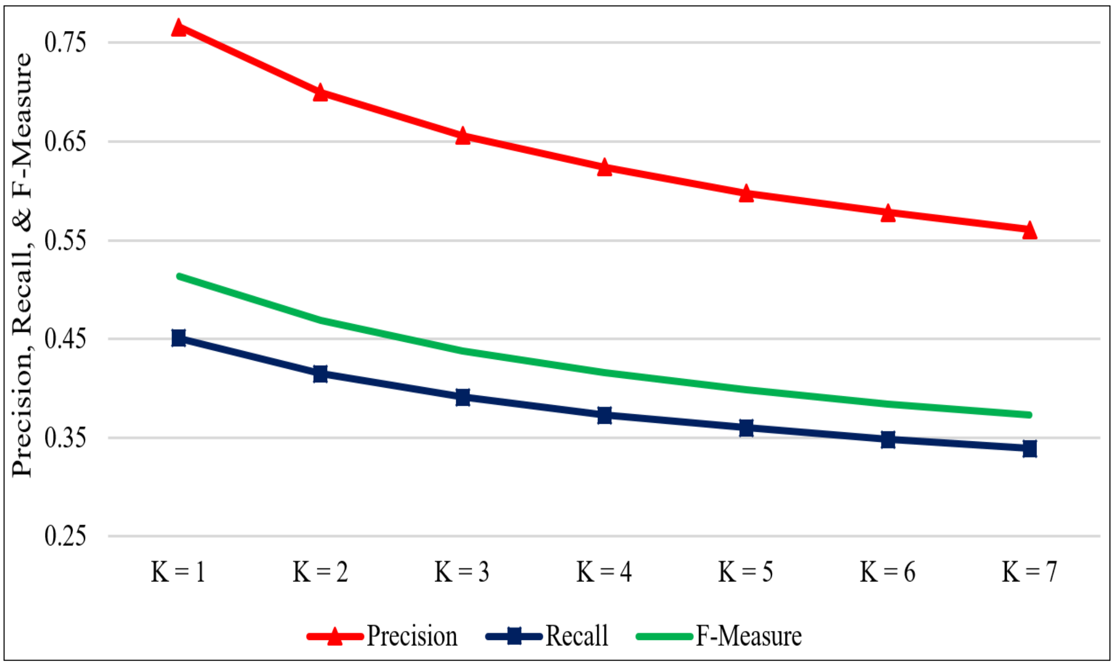


FIGURE 3.10: Performance of QuPiD Attack under Different K values of IBk Algorithm.

3.5.1.4 Performance of QuPiD Attack with Artificial Neural Network

Apart from the selected five best classification algorithms, we also tested the performance of the QuPiD attack with Recurrent Neural Network (RNN) one of the famous algorithms of ANN used for a sequence to sequence translation. In RNN, we used Long Short-Term Memory (LSTM) architecture that uses a feedback mechanism for better classification. LSTM can process both single data points and sequence data. Initially, we tested the performance of the QuPiD attack under LSTM with the same dataset and attributes (Feature Vector) i.e. topic score. However, the results show that the performance of the QuPiD attack with LSTM under topic score Feature Vector was a complete disappointment both in terms of *precision* (0.211) and *recall* (0.135), although we had hoped for better results. Upon investigation, it was found that LSTM misclassified most of the queries due to overlapping classes in the prediction model. We, therefore, consider textual data (query string) instead of the topic score Feature Vector to get better results. We used the "Word2Vec" tool from "Affective Tweets" [145] weka package to produce Word Embedding [146] and then used Dl4MlpClassifier package [147] to train the model using LSTM. In this research, we conducted experiments with 5, 10, 15,

Epochs	Precision	Recall
5	0.936	0.432
10	0.95	0.457
15	0.936	0.462
20	0.936	0.463
25	0.936	0.462
30	0.934	**0.467**
50	0.932	0.459
100	0.934	0.466

TABLE 3.7: Performance of QuPiD Attack with Artificial Neural Network.

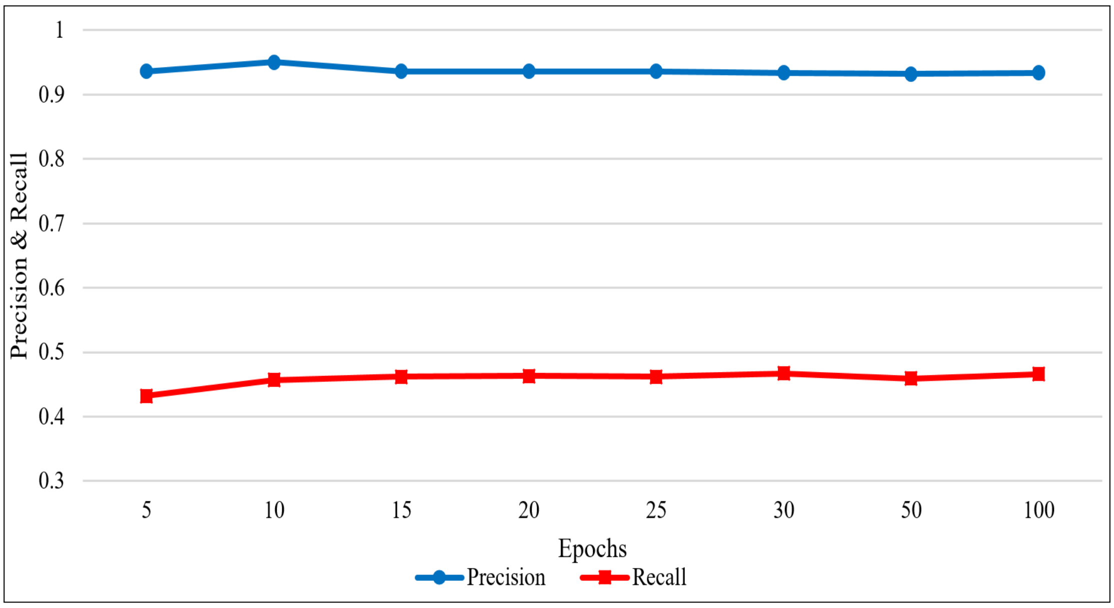


FIGURE 3.11: Performance of QuPiD Attack with Artificial Neural Network.

20, 25, 30, 50, and 100 epochs to observe the performance of the model. (Other parameters are mentioned in Table 3.2).

The results show that the performance of the QuPiD attack with LSTM is far better with textual data as compared to the topic score in terms of precision and recall and it is even close to the results of our best classifier i.e. IBk. With the same textual data, IBk gave 0.49 *recall* with the *precision* of 0.934 while LSTM gave 0.432 *recall* with 0.936 *precision* when the epoch value is set to 5. While the performance of the LSTM model getting better with the increase in the number of epochs. The results of the experiments are shown in Table 3.7 and Fig. 3.11. The maximum *recall* we got is 0.467 with 30 number of epochs. While the *recall* rate for the rest of the epochs fluctuates between 0.432 and 0.463. From the results, it is concluded that the performance of LSTM is slightly inferior then IBk and it can be improved by the fine-tuning of other parameters and functions used to build the prediction model.

3.5.1.5 Conclusion of Classifier Evaluation

The whole exercise was performed to find a suitable classifier which that is compatible with the nature of the dataset. Given the results, it turns out that IBK, Bagging, J.48, and XGBoost are the most suitable candidates for successful QuPiD Attack. However, IBK gave maximum *precision*, *recall*, and *f-measure* followed by XGBoost with over 48% *f-measure* score. Moreover the results of RNN with Long Short-Term Memory (LSTM) are also promising and can be improved with fine tuned parameters and functions. Based on results, we will therefore discuss the results of IBK in the rest of the evaluations in Section 3.5.

3.5.2 Significance of Feature Vector

In the field of machine learning, features play an important role in the identification of a certain entity or event. The Feature Vector is a set of different features (i.e., numeric, text, pixels and others depend on the domain of the problem), representing an object or an event. As most of machine learning algorithms are based on statistical formulae, they need most of the problems to be presented in a numeric form. In previous attempts, Petit et al. [2, 46] proposed the SimAttack based on Dice's coefficient for finding the similarity between the user profile and the query [114]. SimAttack uses the query attribute to find similarity between the user profile and query using Dice's coefficient. Peddinti and Saxena [14] used "query" and "date" attribute as a Feature Vector. They converted date attribute into a numeric attribute using "WEKA DATE" tool. For the query, Peddinti and Saxena used "String To Word Vector" preprocessing filter provided by WEKA. Gervais et al. [111] proposed a Linkage function for finding the similarity between query

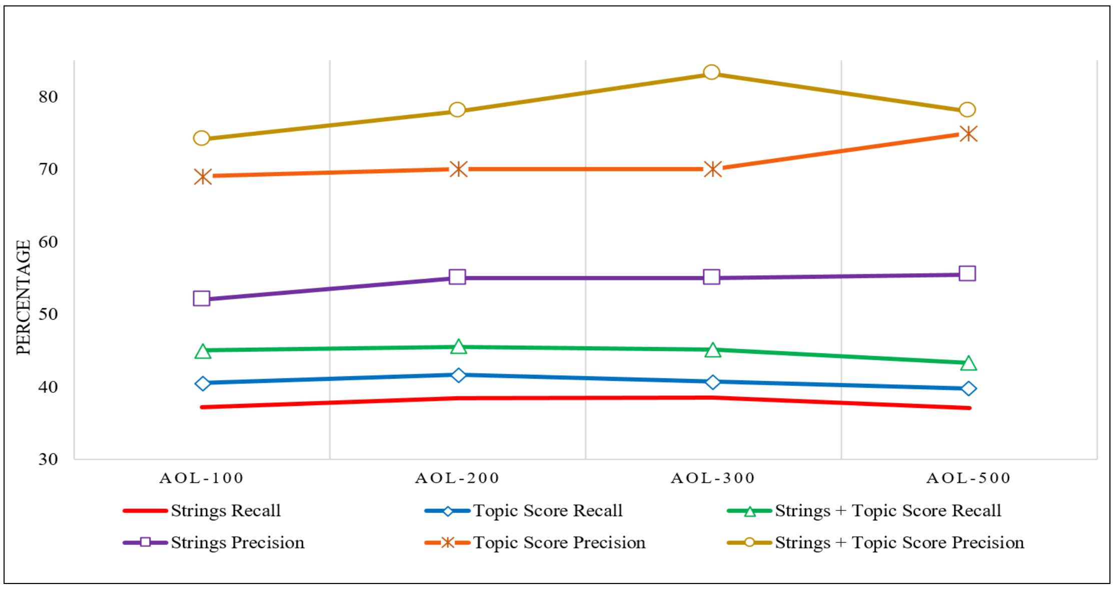


FIGURE 3.12: Ability of Feature Vector: Comparative analysis of *precision* and *recall* of Machine Trained with three different Feature Vectors: Strings, Topic Score, and Strings + Topic Score.

event and history of the user. The Feature Vector used for similarity comprised user identity, query time, query string, ranked list of SERPs (search engine results pages) and links of pages clicked by the user from SERP (search engine results pages).

In previous works [2, 14, 45, 46], all attributes available in AOL dataset were used for breaching user privacy. For this research, a new Feature Vector (Topics Score) was used, adding a new dimension to query identity in the features inventory. Topic Score is composed of score of query in 10 major topics acquired from uClassify Service. An experiment was conducted on different datasets (AOL-100, AOL-200, AOL-300, and AOL-500) to compare the performance of the classification model over different data sets trained with (i) Strings (Query Strings), (ii) Topic Score, and (iii) Both Query Strings and Topic Score. The results show that the Topic Score *recall* rate is much better as compared to Strings. However, we can get better results while using them together as shown in Fig. 3.12. With the ability to de-anonymize 52% to 55% users and their queries on the average with the accuracy of up to 83% in some cases, Topic Score proved itself a potential candidate for the future research. We will discuss the results of models build on Strings + Topics Score in the rest of the evaluation section (Section 3.5).

3.5.3 Impact of Group Size

The behavior of group size is taken as an important parameter in many previous studies. Petit et al. reported that the probability of query association with the correct user is inversely proportional to the number of users in the system [46]. The same observation has also been reported by Khan et al. [72, 124]. In this case, the query association probability divided among the number of users appeared in a session window. Therefore, in order to study the influence of group size over the query association, four group size 3, 6, 10, and 20 were considered. While in order to avoid random error, five experiments are conducted for each group size but with different users. Fig. 3.13 portrays the average impact of group size in terms of *precision* and *recall*, while Fig. 3.14 portrays the standard deviation in *precision* and *recall*.

For the group size of 3 users, IBK associates 36.5% queries to the real user with the *precision* of 86.58%. For the group size of 6 users, IBK associates 35.7% queries to the real users with a *precision* of 76.1%. Similarly, for the group size of 10 users, IBK associates 35.5% queries with the *precision* of 56.32%. For the group size of 20 users, IBK associates 21.62% queries with the *precision* of 94.88% to the original user. The drop of *precision* at a 10 user' group size shows the influence of the ProQSim Effect (Section 3.6). Although the *precision* fluctuates between 55% and 90% for different group sizes, *recall* shows a somewhat nonlinear decline behavior. This behavior of *recall* with an increase in group size indicates that a bigger group offers better security. However bigger group size is not feasible for PIR protocols as it introduce notable delay to result retrieval process.

3.5.4 Impact of Number of Queries

This experiment is conducted to study the impact of the number of queries submitted by *UoI* on his/her privacy in a session using PIR protocols. It is conducted with a group of 20 users with a varying number of queries between 10, 13, 15, 18,

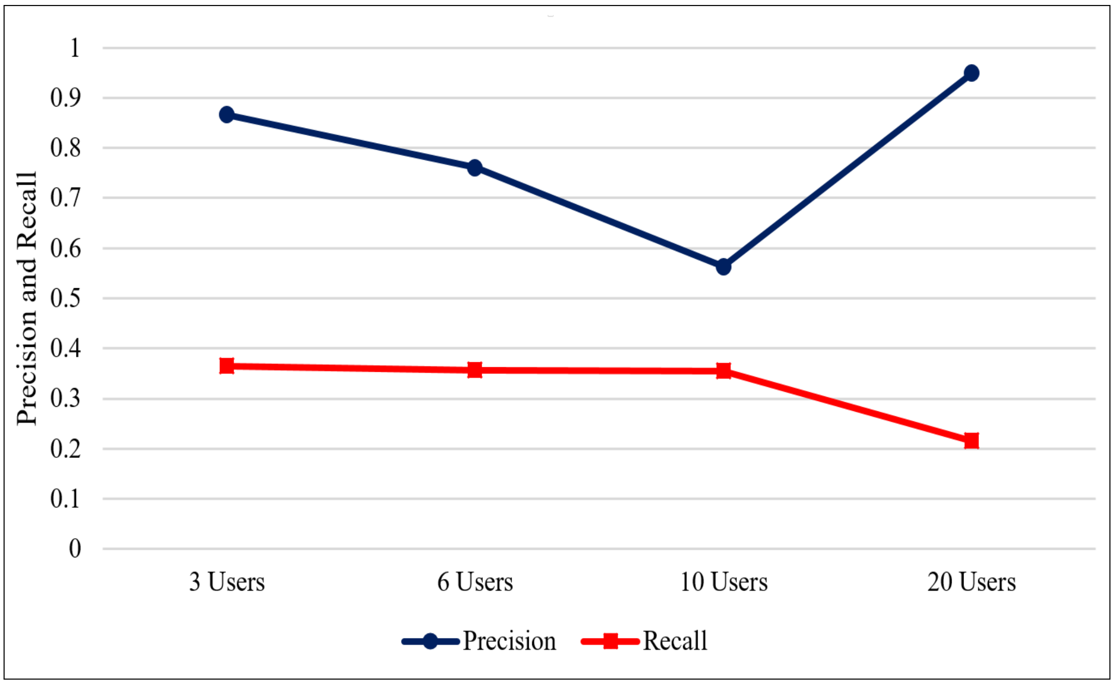


FIGURE 3.13: Impact of Group Size (Average). The Blue Color line represents the Average *precision* Value for Different Number of Users (i.e. 3 Users, 6 Users, 10 Users, and 20 Users). While the Red Color Line represents the Average *recall* Value for Different Number of Users (i.e. 3 Users, 6 Users, 10 Users, and 20 Users).

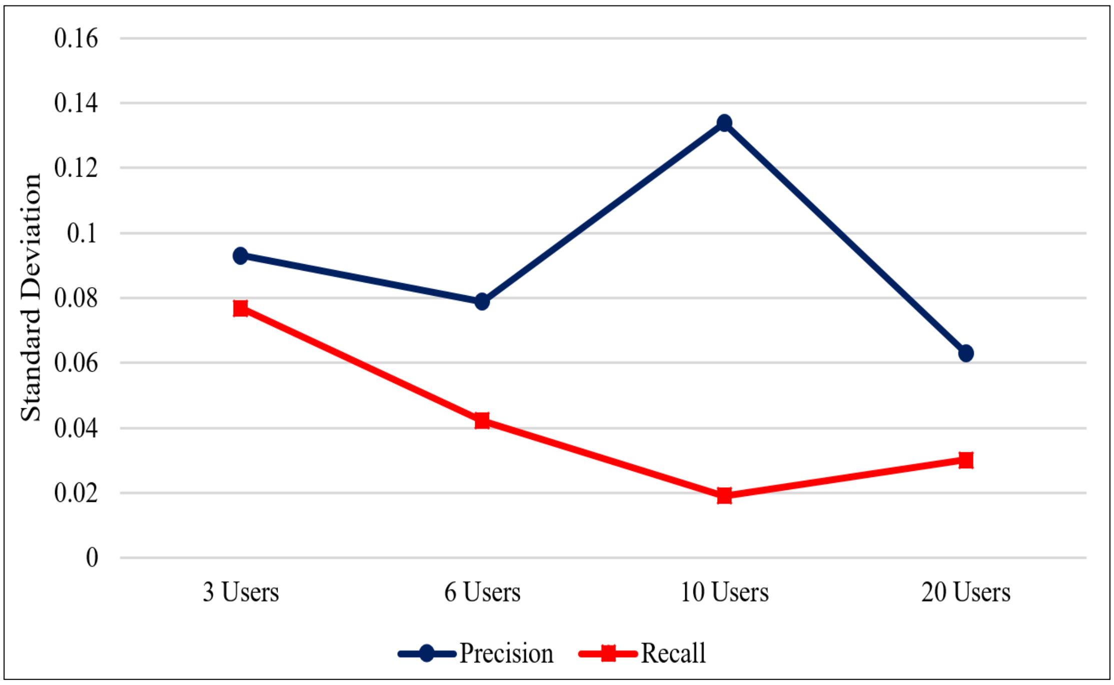


FIGURE 3.14: Impact of Group Size (Standard Deviation). The Blue Color line represents the Standard Deviation in *precision* Value for Different Number of Users (i.e. 3 Users, 6 Users, 10 Users, and 20 Users). While the Red Color Line represents the Standard Deviation in the *recall* Value for Different Number of Users (i.e. 3 Users, 6 Users, 10 Users, and 20 Users).

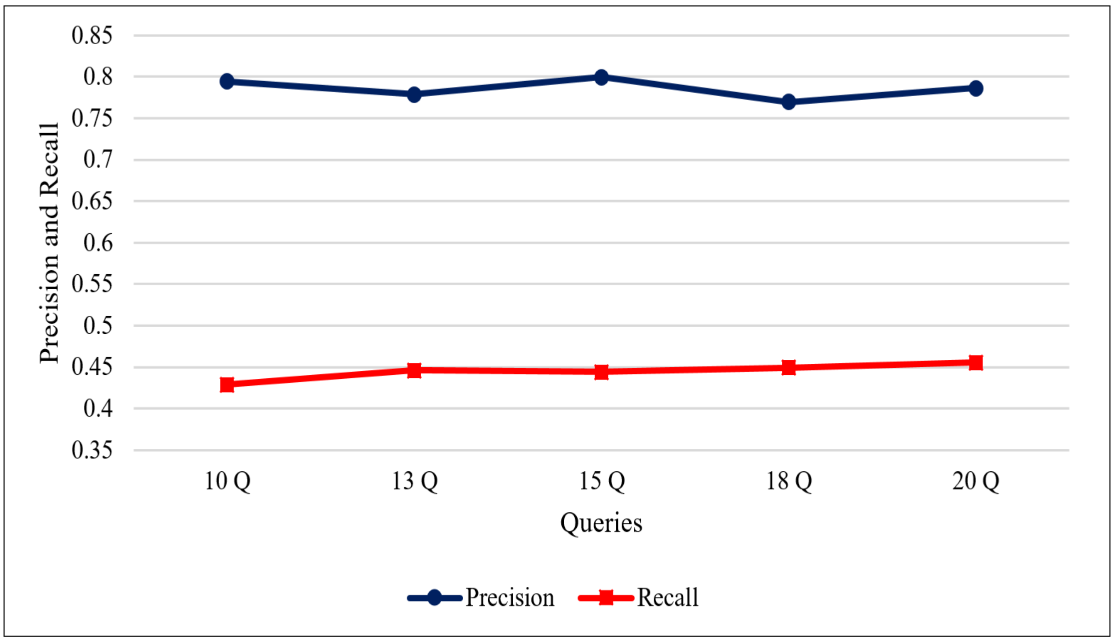


FIGURE 3.15: Impact of Number of Queries (Average).

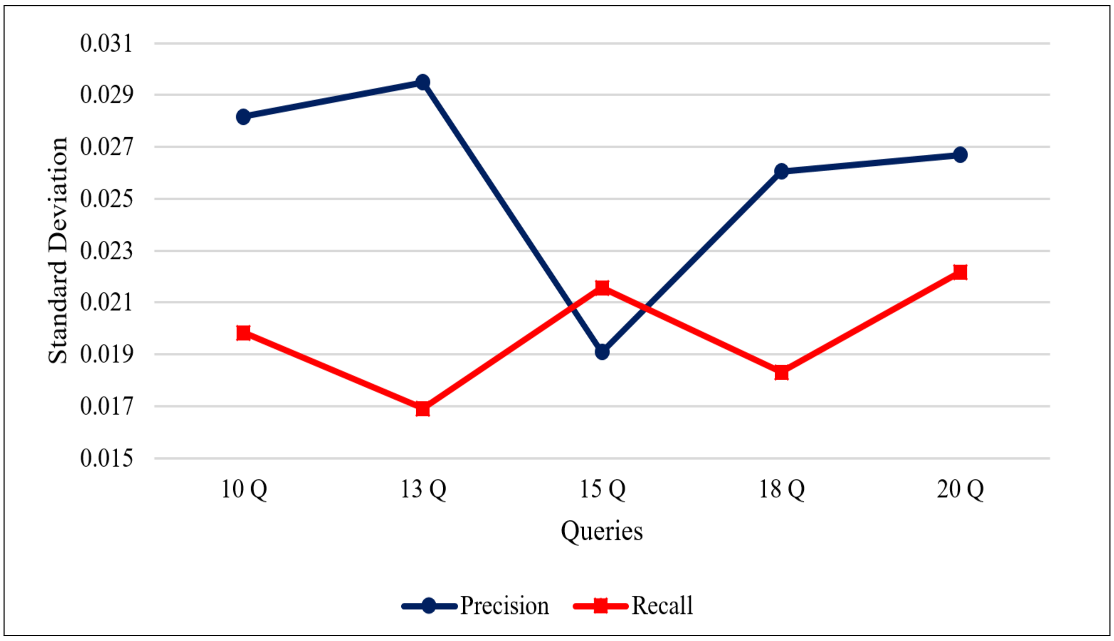


FIGURE 3.16: Impact of Number of Queries (Standard Deviation).

and 20. While in order to avoid the random error, five experiments are conducted for each selected number of queries but with different queries each time. Fig. 3.15 portrays the average impact of number of queries in terms of *precision* and *recall*, while Fig. 3.16 portrays the standard deviation of *precision* and *recall*.

The results show that the rate of *recall* increases with the increase in the number of queries while a slight drop is recorded in *precision* rate with the increase of the

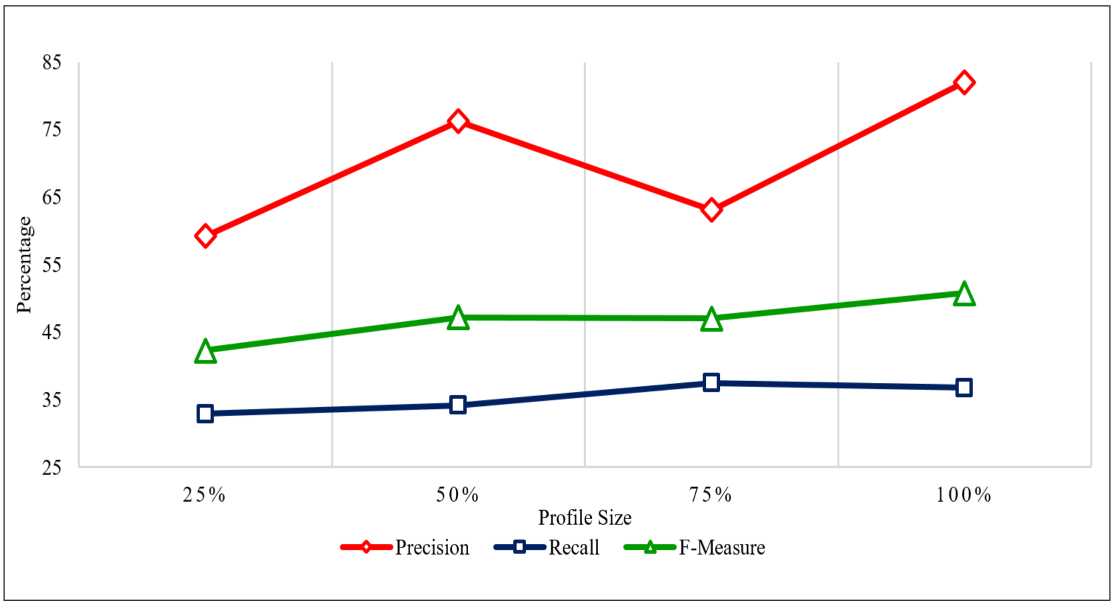


FIGURE 3.17: Comparison of Four subsets of Training Data.

number of queries. In a nutshell, the number of queries has a very significant effect on the association of queries with a real user. However, the group size of fewer than 20 users might bear different results.

3.5.5 Impact of User Profile Size

This experiment was conducted to study the impact of the size of training data on the classification process. As in the previous experiments (Section 3.5.2, 3.5.3, and 3.5.4), the first two months data of AOL dataset was used for machine learning model training and the last month data was used for testing [14, 46, 72, 111]. Consequently, to conduct this experiment, we randomly selected a group of 20 users from AOL-100. Out of this training dataset, we created four subsets. The first subset contained 25% randomly selected queries of each user submitted in the first 61 days. Similarly, the second and third subset contained 50% and 75% randomly selected queries of each user submitted during the same period. While the whole 2 months data of the selected user was used as a 100% subset. All four subsets were used as training data and last month's data was as testing data.

Fig. 3.17 shows the comparison of all four subsets in terms of *precision* and *recall*. The result shows that the rate of *recall* is getting better with the increase in the size of training data except for 100% profile. The possible explanation for this behavior of 100% profile is the presence of ProQSim effect. QuPiD Attack associates more than 30% queries correctly in all scenarios. The pattern of *recall* improves with the increase in profile size except for 100% profile, which seems to be slightly affected by the ProQSim effect. On the basis of these results, it is concluded that the accuracy of the classification model is mildly dependent on the size of the training data. However, this experiment is conducted on limited dataset (3 months data) which is also fairly old. Moreover, we cannot restrict the users interests into 10 major classes.

3.5.6 Impact of Session Window Size

The experiment is conducted with 5 groups, whereby each group was composed of 20 users. Every user of the group submitted 20 queries in each hour of the day. The aim of this experiment is to study the effect of the session window size on the user de-anonymizing process and the effectiveness of the QuPiD attack during different times of the day. For each hour, we used the time-based session window. In the time-based session window, a single day is divided into 24 session windows and each window represents an hour of the day. The details of time-based session window is available in Section 3.3.

Fig. 3.18 depicts the impact of the session window size in the process of associating queries to the original users. If further shows that QuPiD attack is successful both in terms of *precision* and *recall* with respect to the user session window size. From 1 am to 6 am the rate of *recall* is getting better with the decrease in traffic as compared to the rest of the day. The reflection of this observation can also be seen in the rate of *precision* during the above stated period. The maximum *recall* rates achieved by the QuPiD Attack are 45.7% for the queries related to 6 am timespan. After 6 am, a decline in *recall* rate is observed. After 10 am the rate of *recall* becomes somewhat steady. From 10 am to 11 pm the *recall* rate fluctuates between 28% and 34%, respectively. Consequently, the results validate the fact that the rate of *recall* and *precision* both are significantly dependent on the size

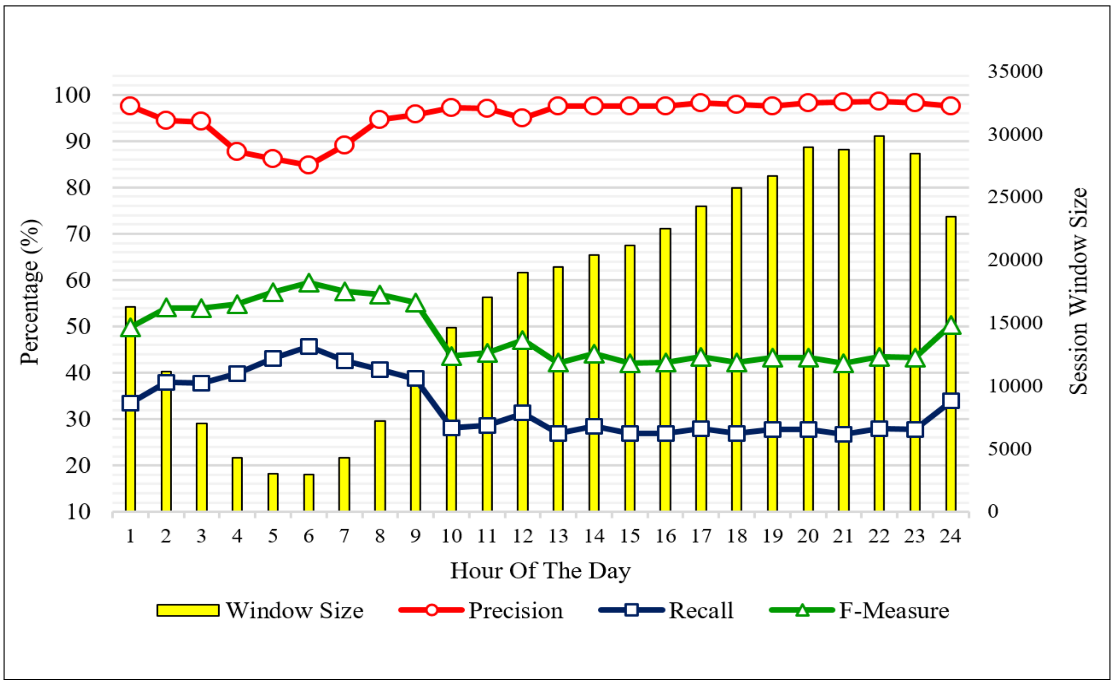


FIGURE 3.18: Impact of Session Window Size.

of the window. These experiments also have traces of ProQSim effect shown at 8 am and 12 am. The details of ProQSim effect are discussed in section 3.6.

3.6 ProQSim Effect

This section contains detailed discussion about the discovery of an effect found during experimentation stage. During experimentation, we discovered unexpected behavior of the proposed QuPiD attack in multiple experiments, greatly affecting the rate of *recall* and *precision*. We referred to this behavior as ProQSim (Profile to Query Similarity) Effect. In order to understand and study this behavior at data level, we conducted experiments with 5 groups (each group contains 20 users randomly selected from the AOL-100 dataset) simulated with same day traffic. The results of these experiments in terms of *precision* and *recall* are shown in Fig. 3.19 and Fig. 3.20, respectively.

The *precision* and *recall* rate of all five groups (each group contains 20 users randomly selected from the AOL-100 dataset) showed an unsteady behavior on various points. Rendering it difficult to draw conclusions, especially from the

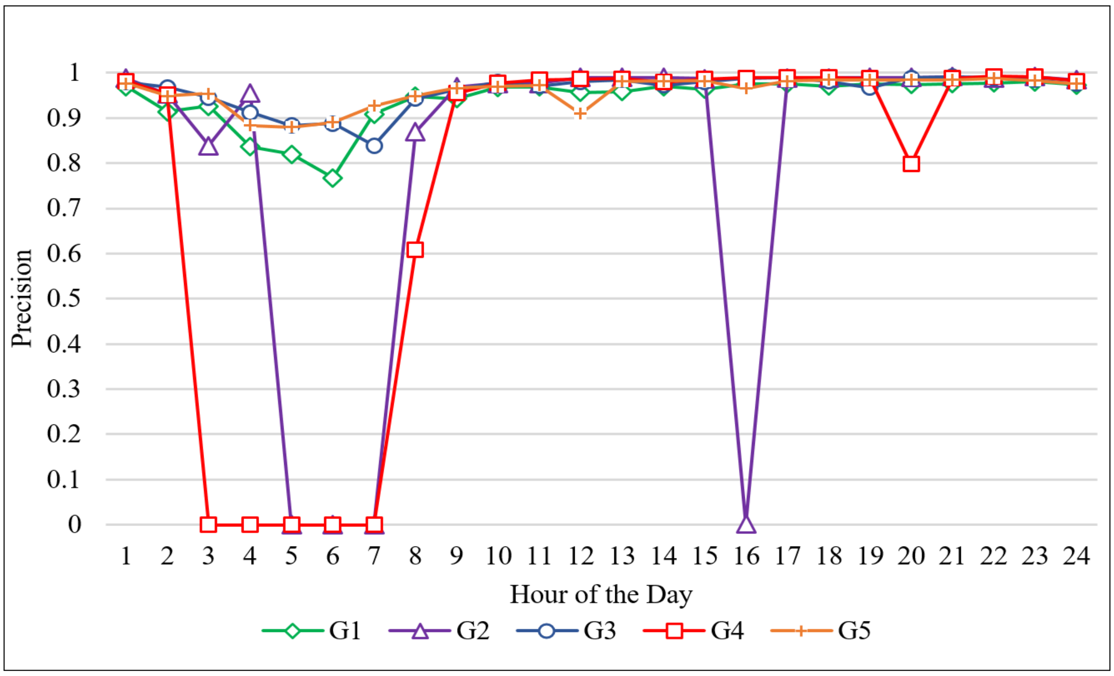


FIGURE 3.19: *Precision* of 5 Groups, Illustration ProQSim Effect.

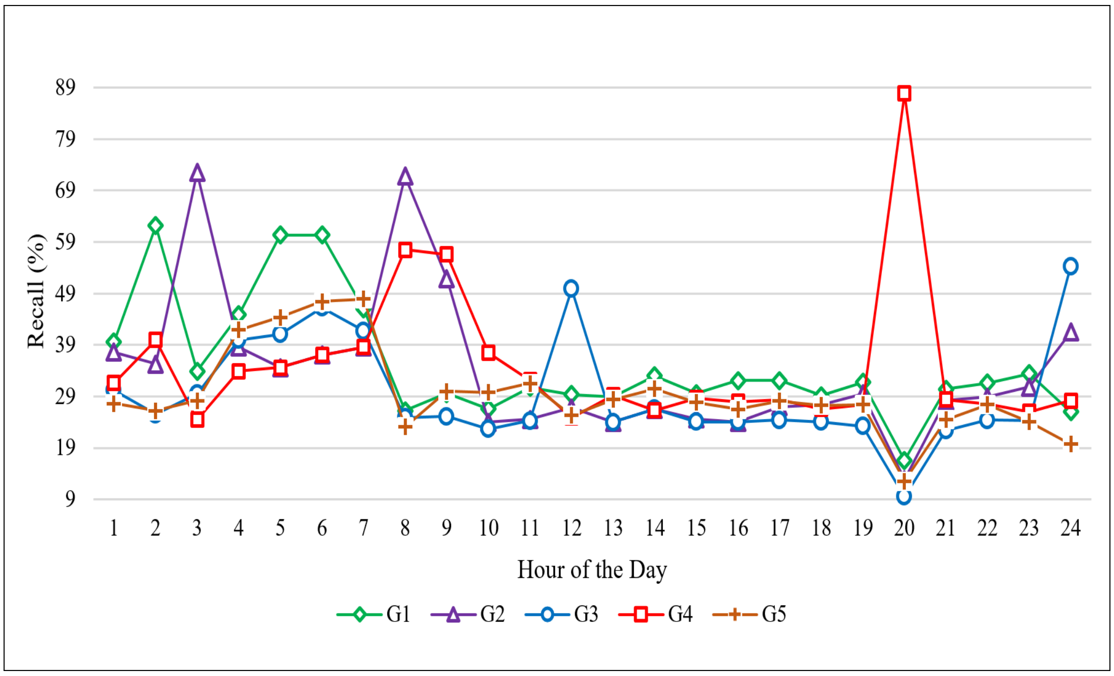


FIGURE 3.20: *Recall* of 5 Groups, Illustration ProQSim Effect.

point of view of *recall* rate where the fluctuation is recorded in results. This sort of irregular and strange behavior is recorded at most of the positions for all five groups as shown in Fig. 3.19 and Fig. 3.20 in terms of *precision* and *recall* respectively. To find the reason behind this strange behavior, we conducted a data level analysis and found three factors responsible for this behavior.

i. The first reason was the lack of traces of incoming query in the user profile. It showed that the user profile on which the machine is trained (training data) is not rich enough to cover all interests of the user. For example, in the user profile all queries are related to either health or sports but in the testing data, the user queries are changed to arts or business-related queries, which are also real queries of the user. In such a case, the traces of the new queries will not be present in the users training profile and when testing, the machine, they will be wrongly classified as other user queries. This is due to the limited dataset; since, as mentioned earlier, the machine is trained only on two months of data representing only a small portion of the user's interests.

ii. Similarly, the second reason for the irregular behavior is the immense difference between the user's training profile and their own testing profile. This is due to the limitation of both the dataset and the Feature Vector, as the dataset represents only a small portion of the user's interests. The Feature Vector classifies the user profile into ten major topics, which cannot encompass all the user's interests.

iii. The third reason for this irregular behavior is the presence of more than one similar user profile. This limitation is connected to a Feature Vector assigning the user profile to ten major classes, e.g., causing features to be similar.

To support the founded responsible factors discovered, a pixel diagram of cosine similarity between training and testing of the same users at a different hour of the day is shown in Fig. 3.21. Likewise, the pixel diagram of Cosine similarity between the users' training profiles are shown in Fig. 3.22.

Each pixel of Fig. 3.21 shows the Cosine similarity [113] between the history of the user's (training profile) and the set of queries submitted by the same user, using any PIR protocol (testing data) in 24 hours. Green color represents the most similarity while red color represents the least similarity. Fig. 3.21 shows the

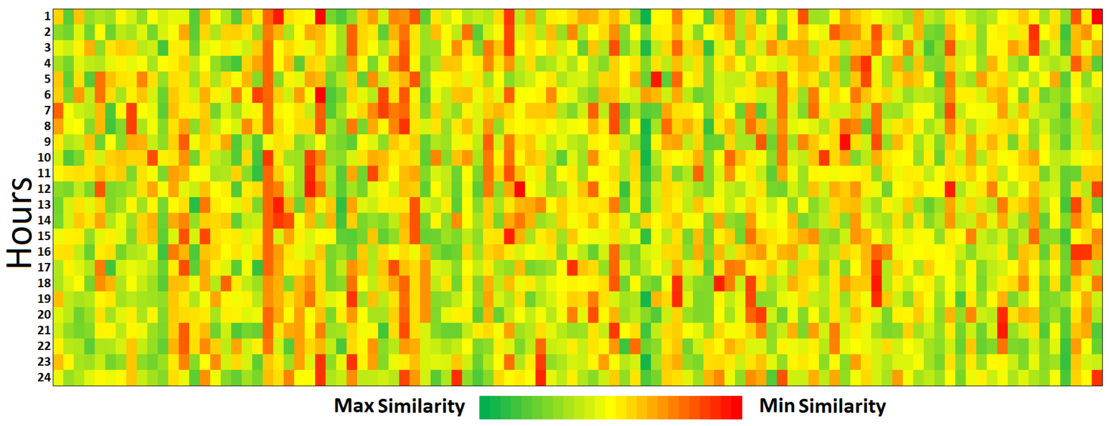


FIGURE 3.21: AOL-100 Training and Testing Data Cosine Similarity Pixel Map.

cosine similarity between the training profile and testing profile of all selected users plotted in 24 hours. The figure is mostly populated with yellow and red pixels, which shows that in many cases, the similarity between training and testing profiles of the user is less than 50%. This variable and less than 50% similarity (in many cases) is one of the reasons for irregular behavior of the Adversarial Model (QuPiD Attack). Similarly, red pixels show complete dissimilarity between training and testing profile of the users. Therefore it is safe to conclude that users profile is not rich enough to encompass all users' interests.

Similarly, Fig. 3.22 represents the cosine similarity between histories of the user (training profiles of all user) with each other. This figure is mostly populated with yellow and red pixels showing that user profiles are dissimilar. However, the presence of mild green and green pixels shows that the profile of some users is closely similar to profile other selected users. These similarities in the users' profile also influence the results seriously.

3.7 Conclusion

In this chapter, we present QuPiD Attack: a machine learning based attack that quantifies the level of protection provided by the popular PIR protocol UUP and is able to break its privacy. The QuPiD attack uses a classification algorithm and

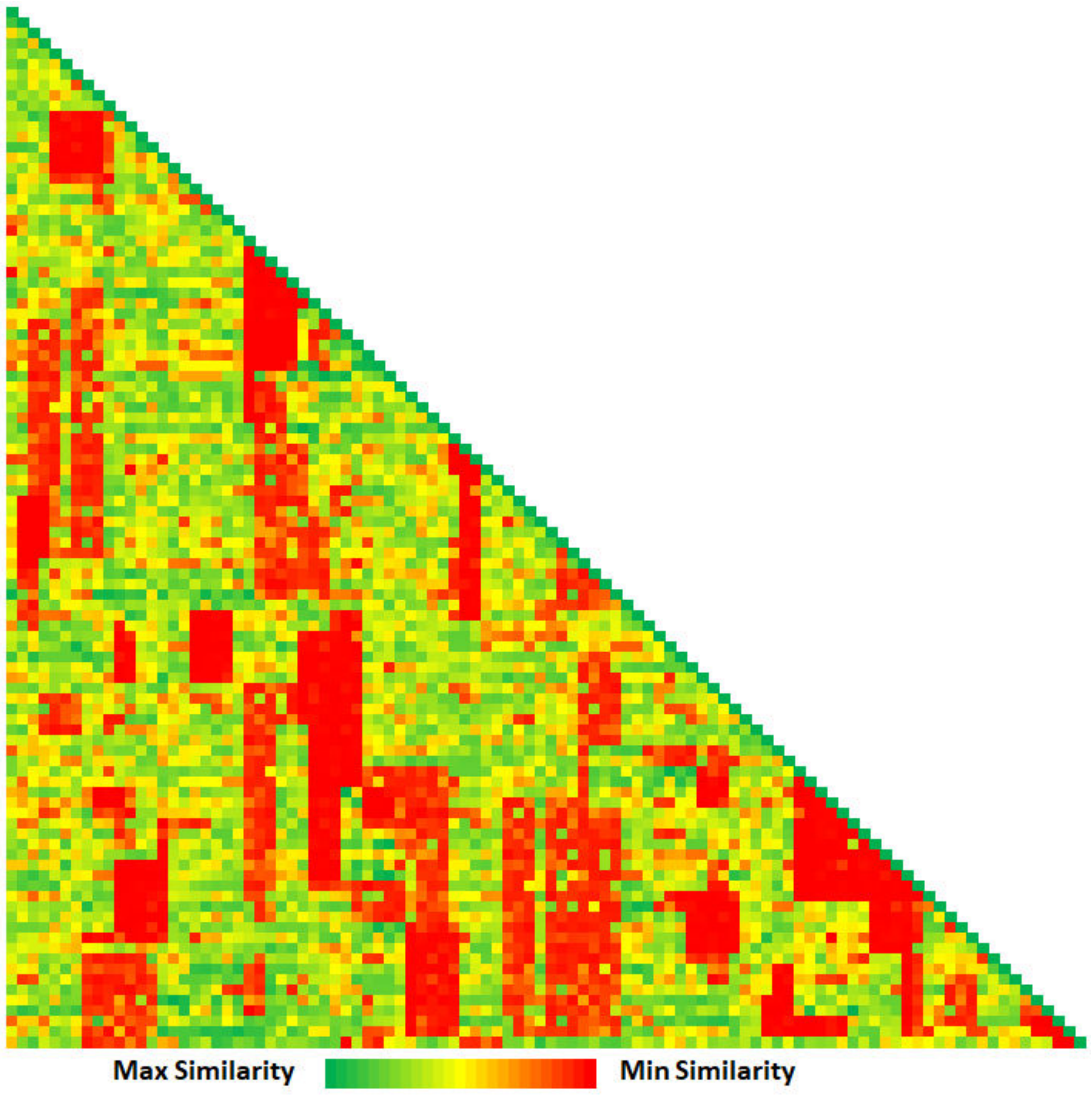


FIGURE 3.22: AOL-100 User to User Cosine Similarity Pixel Map.

the history of the user to classify an incoming query. We empirically showed that using user's real previous profile and the adverse web search engine can successfully breach the user's privacy provided by PIR protocols. For classification, unlike previous research [14, 46, 111], a new "Topic Score" Feature Vector was used with query strings. This research was also the test case of "Topic Score" Feature Vector, which presented 95% *recall* with above 80% *precision* in some cases, proving it to be the reliable and potential candidate for future studies. We studied the level of privacy provided by PIR protocols under varying parameters and multiple scenarios. The experiments included evaluation of classifier, evaluation of Feature Vector, the impact of group size, the impact of the number of queries, the impact of profile size, the impact of session window size, and peak and valley time analyses.

For the selection of the best classification algorithm, we conducted experiments with ten classification algorithms from different families including Tree-Based, Rule-Based, Lazy-Learner, Meta-heuristic family and Bayes family. The results showed that IBK is the most appropriate algorithm if the "strings + categories" Feature Vector is used. We also tested the performance of the QuPiD attack with Artificial Neural Network (ANN) however, the maximum *recall* recorded was 0.211 with learning rate of 0.2 which is poor as compere to the *recall* of rest of selected classifiers. In order to evaluate the impact of group size over the user's privacy, we conducted experiments with the group size of 3, 6, 10, and 20 users. The results showed that a bigger group size offers more privacy to the user. Similarly, in order to evaluate the impact of training data size, we conducted experiments with 25%, 50%, 75%, and 100% profile size of the available data. It was found that the increase in profile has a mild effect on the disclosure of the user's privacy. However, this experiment confirms the obvious fact that a bigger user profile size is more prone to maximum privacy disclosure. In the next experiment, we evaluated the effect of a number of queries submitted by the user in a session over a user's privacy disclosure. The results showed that the accuracy of QuPiD attack is almost the same in the revelation of user queries. In the last experiment, we evaluated the performance of the QuPiD attack during different times of the day. In other words, this experiment shows that at what time of a day (24 hours), the user is more vulnerable to privacy exposure. It was found that the user is more vulnerable to privacy attack between 1 am and 9 am.

During the experimentation, we discovered an unexpected unsteady behavior of QuPiD attack that was captured in the *recall* rate of the attack. We referred to this behavior as the ProQSim (Profile to Query Similarity) Effect. Upon investigation, it was found that this behavior of the QuPiD attack was mainly due to two reasons. (i) The first reason was the lack of traces of the incoming query in the user profile. (ii) The second reason was the presence of more than one user profile that was almost similar. The reason behind these major issues is the limited user profile. However, as compared to the SimAttack [46], the QuPiD attack is still more robust due to the "Topics Score" Feature Vector.

Chapter 4

PEM: A Privacy Exposure Minimizing Technique

Privacy is a sensitive issue from the user's point of view. Unfortunately for web search engines, user privacy is just a behavior tracking and used for multiple purposes including profit. To address the issue of privacy infringement while using WSE, researchers have proposed several techniques and alternatives. These techniques can be classified into four major classes i.e., user anonymizing networks, profile obfuscation, private information retrieval (PIR) protocols, and hybrid techniques. The focus of this thesis is to test the performance of PIR protocols in terms of privacy. In the previous chapter, we proposed QuPiD attack to test the performance of PIR protocols and it was found that PIR protocols cannot provide satisfactory privacy to the user in case of QuPiD attack. Therefore, we propose a novel mechanism for query placement in the existing private information retrieval solutions that will enhance the privacy of the user. We propose a PEM (privacy exposure measure), which is a technique that minimizes the privacy exposure of the user while using the PIR protocols.

In this chapter, we introduce PEM (privacy exposure measure), a privacy exposure estimation module for private information retrieval protocols to reduce the user's privacy exposure. PEM assesses the similarity between the user's profile and the query before posting to the WSE and it assists the user in avoiding privacy exposure. As the web search engine already has the user's real profile (history), we use a privacy exposure threshold value in PEM that will facilitate the user to avoid QuPiD attack and thus avoid further privacy exposure. This way the success rate of the user identification remains low in the case of a QuPiD attack. We used Jaccard similarity [148], Cosine similarity [113], and Euclidean distance [149] for similarity calculation, in order to find a suitable similarity measure. For privacy evaluation, we used QuPiD attack, Profile similarity, Kullback-Liebler (KL) divergence [150], and Cross-Entropy loss [119].

In this chapter, we will discuss the following: 1. detail design of PEM, 2. proposed Algorithm of PEM, 3. experimentation setup, 4. similarity mechanisms used in PEM, and 5. performance evaluation metrics. We will conclude this chapter with conclusions and some recommendations.

4.1 Detail Design of Privacy Exposure Measure

In this section, we present PEM (privacy exposure measure), a technique that minimizes the privacy exposure of user while using the PIR protocol. In PIR protocol, the generic steps for private information retrieval are [40]:

1. Group creation/joining;
2. Queries shuffling among group members;
3. Query submission to WSE;
4. Results Dissemination.

Due to the shuffling of queries between group members, WSE cannot identify the real users of the queries. However, WSE can build a classification model using the user's previous profile and supervised learning algorithms and can classify the incoming queries with adequate accuracy. In order to minimize the privacy

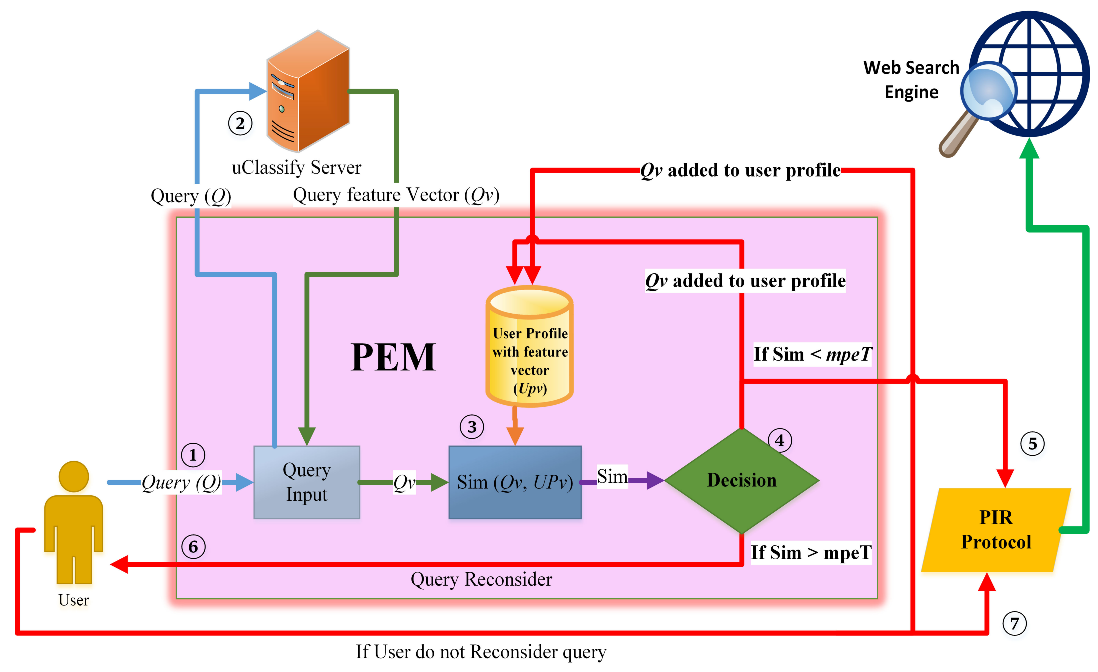


FIGURE 4.1: Operation of PEM.

exposure in case of QuPiD attack (or any other machine learning attack), we propose PEM before queries shuffling step. PEM measures the similarity between the user profile and the new query. If the similarity between the user profile and the query is above the threshold, the user is asked to reconsider the query. As a consequence, the low similarity between queries and users profile prevents the machine learning attack to classify accurately. The working of Privacy Exposure Measure (PEM) is shown in Fig. 4.1.

For working, PEM needed (i) user maximum privacy exposure threshold $mpeT$, (ii) profile feature vector UPv, and (iii) user query Q. The user is assumed to be equipped with his/her own previous profile UP. The user profile UP contains queries submitted by the user previously (as shown in Equation 4.1). In an initial step, the user is asked to set the $mpeT$ value (in this research we conducted our experiments with three $mpeT$ values i.e., 40%, 50%, and 60%). The $mpeT$ value allows the user to set his/her desired privacy exposure such as 40% $mpeT$ value means 60% user profile will remain hidden. For profile feature vector UPv, uClassify service is consulted. uClassify is a web-based service that provides a score of each query in 10 classes. The collection of user queries with the feature vector is represented as UPv (as shown in Equation 4.2).

$$UP = \{U_{q1}, U_{q2}, U_{q3}, ...U_{qn}\} \tag{4.1}$$

$$UPv = \{U_{q1}v, U_{q2}v, U_{q3}v, ...U_{qn}v\} \tag{4.2}$$

Algorithm 2 Privacy exposure measure.

```
Input: User profile with feature vector ($UPv$), New Query ($Q$), Max Privacy Exposure Threshold ($mpeT$).
 1: procedure QUERY SIMILARITY($UPv$,$mpeT$,$Q$)
 2:     $Qv \leftarrow$ get feature Vector for ($Q$)
 3:     for $qi \in Uqv$ do
 4:         $qsim \leftarrow Similarity\,(qi, Qv)$
 5:         if $qsim >\ mpeT$ then
 6:             Ask user to reconsider query ($Q$)
 7:             if user reconsider = yes then
 8:                 return ($Q$) to user
 9:             else
10:                 $UPv \leftarrow add(Qv)$
11:                 Send query ($Q$) for shuffling process
12:         else
13:             $UPv \leftarrow add(Qv)$
14:             Send query ($Q$) for shuffling process
```

The working of the PEM Algorithm (Algorithm 2) and depicted in Fig. 4.1. The working of the algorithm is as follow:

1. First, the user is asked to set the maximum privacy exposure threshold value $mpeT$. (Input line)

2. In the second step, the feature vector for users' new query Qv is acquired from uclassify service. (line 2)

3. After acquiring the feature vector, the third step is to find the similarity between new query Qv and user profile queries UPv. To evaluate different similarity calculation methods, we conducted our experiments with Cosine similarity, Jaccard similarity, and Euclidean Distance. The details of similarity calculation methods are discussed in Section 4.2.2. (lines 3-4)

4. After the similarity calculation, in the next step, the query similarity score $qsim$ is checked with maximum privacy exposure threshold value $mpeT$. (line 5)

5. If the value of $qsim$ is less than $mpeT$, the query Q is forwarded for the shuffling process and query vector Qv is added to the user profile UPv. (lines 5, and lines 12-14)

6. If the value of $qsim$ is greater than $mpeT$, the user is asked to either reconsider the query or submit the query without any modification to the shuffling process. (lines 5,6)

   (a) If the user select "***query reconsiders***" option, the query (Q) is sent back to the user for modification. (lines 7,8)

   (b) If user select "***query shuffling process***", the query is forwarded for shuffling process with privacy exposure warning and query vector Qv is added toser profile UPv. (line 7, 10, 11)

For query modification, a user can adopt different methods available such as "generic queries" [77, 151], and "query scrambling" [78]. In order to evaluate the performance of PEM-powered-PIR protocol, we choose UUP, a prevailing PIR protocol. The experiments are conducted using AOL dataset. For the recommendation of suitable similarity, the experiments are conducted with three similarity methods i.e., Cosine, Jaccard, and Euclidean under $mpeT$ values 40%, 50%, and 60%. Similarly, QuPiD attack is then used for privacy exposure evaluation for each setting. The details of the experiments are discussed in Section 4.3.

4.1.1 Time and Space Complexity of PEM

As mentioned earlier, Privacy Exposure Measure (PEM)is proposed as additional privacy step in the already existed Private Information Retrieval (PIR) protocols (such as Useless User Profile (UUP)) [37], therefore PEM-powered-PIR will inherit the time and space complexity from its base protocol. However, based on the algorithm steps, the worst-case running time of only Privacy Exposure Measure (PEM) step is *O(n)* or linear. Similarly the worst case space complexity of Privacy Exposure Measure (PEM) step is also *O(n)*. Where n shows the number of queries present in the user's profile or history. Based on the algorithmic time and space complexity analysis, it is concluded that Privacy Exposure Measure (PEM) added very small delay to the entire private information retrieval process.

4.1.2 User Profile

The user profile contains the queries sent by the user previously with feature vector values acquired from uClassify services. According to the working of Privacy Exposure Measure (PEM), the similarity between each new user query Qv and every query present in the user profile UPv is calculated. If the similarity between all the queries and the new query is less than the threshold value, then the query is forwarded to the Private Information Retrieval (PIR) protocol and query Qv is added to the user profile. If the similarity between user new query Qv and user profile UPv is greater than or equal to the maximum privacy exposure threshold value ($mpeT$ value). Then the new user query Qv is sent back to the user for either to reconsider the query or submit the query without any modification to Private Information Retrieval (PIR) protocol.

4.1.3 Maximum Privacy Exposure ($mpeT$) Value

Maximum privacy exposure ($mpeT$) is the threshold value set by the user for exposure of his/her profile to the web search engine. Although users use Private Information Retrieval (PIR) protocols to ensure their while using web search engines. However, as described previously Private Information Retrieval (PIR) protocols are vulnerable to QuPiD attack and cannot offer maximum privacy. Therefore, with Privacy Exposure Measure (PEM) user is enabled to set his/her maximum privacy exposure and to avoid privacy exposure in case of QuPiD attack.

4.2 Experimental Setup

In this section, we present the experimental evaluation of Privacy Exposure Measure (PEM). We first describe the dataset and the user selection process we used in Section 4.2.1. Then we present the profile and query similarity calculation methods used in this research (Section 4.2.2). In the last section 4.2.3, we present the machine learning attack and performance evaluation methods that we use to assess PEM. To perform these experiments, we used two-third of user queries as user profile (UP) for similarity calculation and training data for machine learning attack. While the rest of the data is used as new queries (Q) of the user in similarity calculation and testing data for machine learning attack. Nine experiments are conducted to test the effectiveness of the proposed modification in the Private Information Retrieval (PIR) protocols with three similarity methods. The effectiveness of the proposed model Privacy Exposure Measure (PEM) with all three similarity calculation methods is evaluated in terms of permissible queries, profile similarity, and machine learning attack. All experiments were conducted on a workstation with a 4.1 GHz intel Core i7 processor and 16 GBs memory.

4.2.1 Dataset and User Selection

We used AOL web search log for conducting the experiments. The AOL web search logs consist of queries submitted by 6.5 million users (approximately) during the three months period from March 2006 to May 2006. The dataset consists of five attributes user ID, query, date and time of the query, clicked content rank, and clicked URL. For experimentation, we only consider user query attribute of the AOL dataset. For each query, the feature vector is obtained from the uClassify service. uClassify is an online service that provides different kinds of text classifiers such as sentiment analysis, topics, language detection, and many others. For the experiments, topics classifier is used that classifies the query string into 10 major topics. The topics are Sports, Arts, Society, Business, Science, Computers, Recreation, Games, Home, and Health. uClassify gives the score of the query in

TABLE 4.1: Score of Query "Heart Failure" from uClassify.

Health	Society	Home	Arts	Games	Business	Computers	Science	Recreation	Sports
0.381	0.161	0.070	0.068	0.062	0.058	0.055	0.054	0.049	0.042

TABLE 4.2: Dataset Properties.

Total selected users	50
Total queries	35835
Total queries used as training data (UP)	23366
Total queries used as testing data (Q)	12469
Max queries by a single user	2490
Min queries by a single user	183
Time duration	3 Months (01 March 2006 31 May 2006)

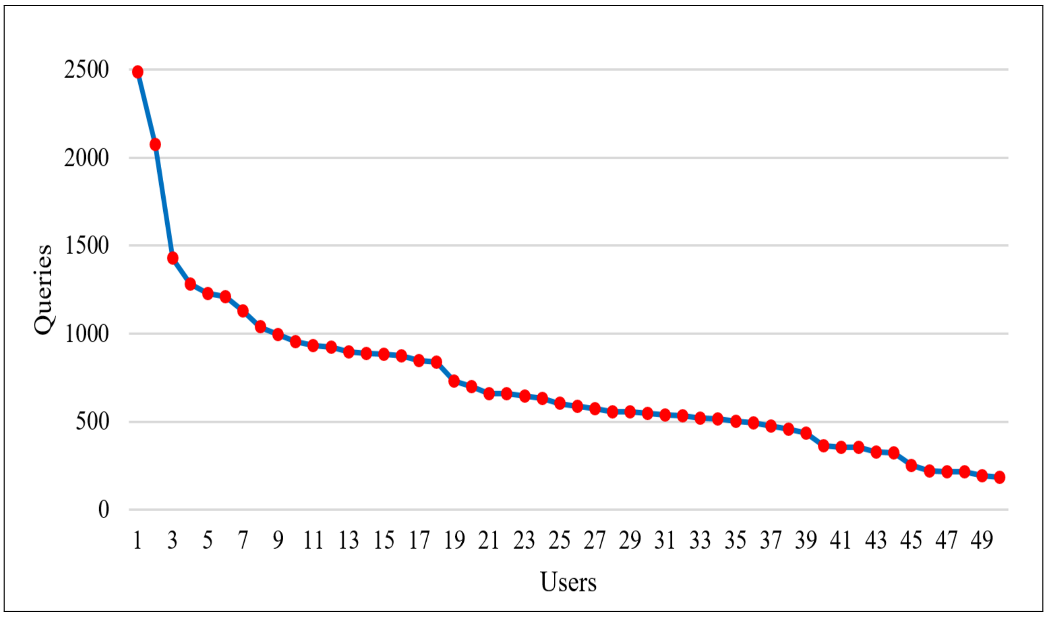


FIGURE 4.2: Number of Queries submitted by Selected User.

previously mentioned topics. For example, for query "Heart Failure", the score for each topic is shown in Table 4.1.

The user selection was made based on the activity of the user during the entire span of the dataset. Instead of concentrating on all users, we randomly choose 50 users who have submitted a minimum 180 during the entire 3 months period for experiments. The query count of selected 50 users during three months is shown in Fig. 4.2. The data of the first two months is used as user profile (UP), while last month data is used as new queries (Q). the summary of the selected dataset is provided in Table 4.2.

4.2.2 Similarity Methods

As discussed earlier, one of the objectives of this work is to recommend a suitable similarity algorithm. Therefore, we used Cosine, Jaccard, and Euclidean methods for similarity calculation. As these methods are applicable to vector data, therefore we model the queries from the user profile (qi) and new query (Qv) as a vector (composed on score acquired from uClassify for both (qi) and (Qv)). The Cosine, Jaccard and Euclidean similarities between two vectors qi and Qv are defined in Equations 4.3, 4.4, and 4.5, respectively.

$$CosineSimilarity(qi, Qv) = \frac{\sum_{j=1}^{n} qij\ Qvj}{\sqrt{\sum_{j=1}^{n}(qij)^2}\sqrt{\sum_{j=1}^{n}(Qvj)^2}} \tag{4.3}$$

$$JaccardSimilarity(qi, Qv) = \frac{\sum_j Min(qij, Qvj)}{\sum_j Max(qij, Qvj)} \tag{4.4}$$

$$EuclideanDistance(qi, Qv) = \sqrt{\sum_{j=1}^{n}(qij - Qvj)^2} \tag{4.5}$$

where, vector qi represents the i^{th} query of the user profile while Qv represents a new query of the user. Similarly, qij and Qvj shows the j^{th} component of the profile and the new query vector.

4.2.3 Performance Evaluation Methods

For the evaluation of PEM-Powered-PIR protocol (UUP protocol with PEM) in terms of privacy, Initially, we used QuPiD attack. As discussed previously, PIR protocols do not allow web search engines to build the user's actual profile. However, our proposed QuPiD attack can break the user's privacy by associating queries to the correct user in an anonymized log. Therefore, in order to evaluate the privacy of PIR protocols with our suggested modification, we used QuPiD attack as privacy quantification model of PIR protocols. For building the classification model, we used UPv (users history) as training data and IBK as classification algorithm. For evaluating the proposed scheme, we simulate the PEM-Powered-PIR protocol with 50 selected users and create the anonymized log. The anonymized log is used as testing data. The performance of the protocol is then evaluated in terms of *precision*, *recall*, and *f-measure*.

Although QuPiD attack is able to evaluate the privacy of the PIR protocols, however, it calculates the success rate based on the queries sent by the user. As the chief goal of the PEM is to minimize the user's profile exposure by restricting the queries whose similarity is greater than the threshold ($mpeT$) value. Therefore, we used Profile Similarity, Cross-Entropy loss and Kullback-Liebler (KL) divergence to calculate the similarity, information loss and divergence between user profiles made with and without PEM. The Euclidean distance is defined in Equation 4.5 while the Cross-Entropy loss and the KL divergence is defined in Equation 4.6 and Equation 4.7 respectively.

$$H(P,Q) = -\sum_{i=1}^{n} (P_i) \cdot log\,(Q_i) \tag{4.6}$$

$$D_{AvgKL}(qi,Qv) = \sum_{j=1}^{10} \left(\alpha_{qi} \times \left(W_j qi \times ln\left(\frac{W_j qi}{W_j}\right)\right) + \alpha_{Qv} \times \left(W_j Qv \times ln\left(\frac{W_j Qv}{W_j}\right)\right)\right) \tag{4.7}$$

where for j^{th} term
$\alpha_{qi} = \frac{W_j qi}{W_j qi + W_j Qv}, \alpha_{Qv} = \frac{W_j Qv}{W_j qi + W_j Qv}$ $and\ W_j = \alpha_{qi} \times + W_j qi + \alpha_{qv} \times W_j Qv$

where j, represents the value of the vector acquired from uclassify. Kullback-Liebler (KL) divergence is used to measure the difference between a probability distribution with a reference probability distribution. In our case, KL divergence is used to measure the difference between the user profiles created with and without (PEM) Privacy Exposure Measure) in various settings (such as the $mpeT$ value of 40%, 50%, and 60%). While Cross Entropy is used to calculate the information loss between the user profiles created with simple PIR (Private Information Retrieval) Protocols and PEM-Powered-PIR protocols in various settings.

Similarly, in order to measure the effectiveness of machine-learning attack (QuPiD Attack) against PEM-powered-PIR protocol, we consider standard performance evaluation metric *precision*, *recall*, and *f-measure*. The mathematical representation of *precision*, *recall*, and *f-measure* is defined in Equation 4.8, Equation 4.9, and Equation 4.10 respectively.

$$precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{4.8}$$

$$recall = \frac{TruePositive}{TruePositive + FalseNegative} \tag{4.9}$$

$$f \cdot measure = 2 \times \frac{precision \times recall}{precision + recall} \tag{4.10}$$

where, *True Positive* and *False Positive* represents the portions of positive and negative cases that correctly classify as positive and negative by the classification model. While *False Positive* represents the portions of negative cases that classify as positive by the classification model.

4.3 Results and Discussion

This section contains the discussion about the results of the experiments we conducted to evaluate the effect of our proposed technique Privacy Exposure Measure (PEM) for privacy-preserved web search. The experiments are conducted with three similarity methods i.e., Cosine, Jaccard, and Euclidean in order to find a suitable similarity calculation method for Privacy Exposure Measure (PEM). These methods are further analyzed using three max privacy exposure threshold ($mpeT$) of 60%, 50%, and 40% for each similarity method. This section divided into three subsections based on different privacy preservation angles offered by Privacy Exposure Measure (PEM). In first section we discussed the performance of Privacy Exposure Measure (PEM) under selected similarity measures (i-e Cosine, Jaccard, and Euclidean) with different $mpeT$ values. In second section we discussed the performance of Privacy Exposure Measure (PEM) in case of QuPiD attack. While in third section we evaluate the performance of Privacy Exposure Measure (PEM) in terms of three privacy measures i-e. Profile Similarity, Cross-Entropy loss and Kullback-Liebler (KL) divergence.

4.3.1 Similarity Measure Evaluation

This section contains the results and discussion about the various experiments conducted to evaluate the performance of PEM with selected similarity measures measure (i-e Cosine, Jaccard, and Euclidean) with three max privacy exposure threshold ($mpeT$) of 60%, 50%, and 40% for each similarity method. Table 4.3 illustrate the number of permissible queries by PEM under the selected similarity techniques (i.e., Cosine, Jaccard, and Euclidean) with a $mpeT$ value of 60%, 50%, and 40%. The results show that Cosine similarity allows 96.31%, 99.60%, and 100% queries under the $mpeT$ value of 40%, 50%, and 60% respectively. Similarly, Jaccard similarity allows 33.50%, 49.71%, and 67.80% queries under the $mpeT$ value of 40%, 50%, and 60%, respectively. While Euclidean distance permit 20.40%, 25.61%, and 35.58% queries under the $mpeT$ value of 40%, 50%, and 60%, respectively. These figures clearly show that the Euclidean distance method outperformed other selected similarity methods (i.e., Cosine and Jaccard) by allowing a small number of queries, thus provide comparatively better privacy. Jaccard similarity, on the other hand, provides mild privacy by allowing more queries than Euclidean distance under each max privacy exposure threshold. Fig. 4.3 shows the permissible queries by Privacy Exposure Measure (PEM) with various settings and permissible queries without Privacy Exposure Measure (PEM).

These results show that PEM is more sensitive to the profile exposure as compere to the UUP protocol. In case of PEM, user profile is more obfuscated as compared to the user profile created using UUP and thus offer more privacy to the user. Although PEM permit few queries and also increase the computational cost of the protocol, however it also empowers UUP to offer more privacy to the user.

TABLE 4.3: Permissible Queries by PEM under Selected Similarity Methods and $mpeT$ Values.

Similarity Method	$mpeT$	Permissible Queries	Permissible Queries %
Cosine Similarity	40%	12009	96.31%
	50%	12419	99.60%
	60%	12469	100%
Jaccard Similarity	40%	4177	33.50%
	50%	6198	49.71%
	60%	8454	67.80%
Euclidean Distance	40%	2538	20.40%
	50%	3193	25.61%
	60%	4436	35.58%
UUP	-	12469	100%

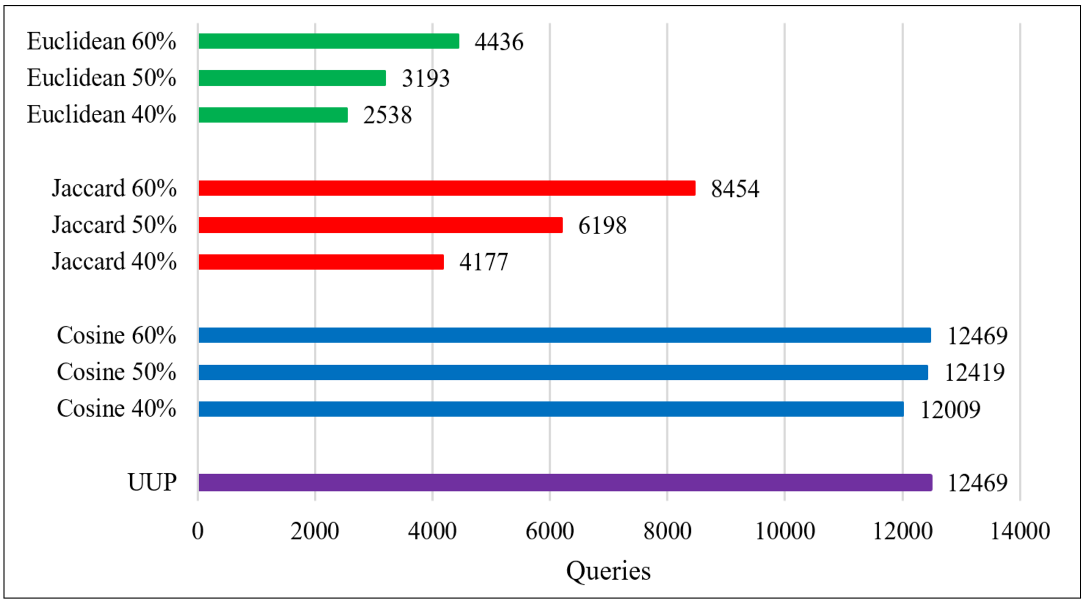


FIGURE 4.3: Permissible Queries by PEM under Selected Similarity Methods and $mpeT$ Values.

4.3.2 Robustness of PEM Against QuPiD Attack

As one of the major objective of this research, we proposed QuPiD attack for the privacy evaluation of PIR protocols. This section contains the results and discussion about the experiments conducted to evaluate the robustness of UUP and PEM against QuPiD Attack.

As mentioned previously, we trained our classification model on UPv (history of selected users) and supplied the anonymized log that contains new queries (Qv) as testing data. Fig. 4.4 shows the efficiency of the model with various testing

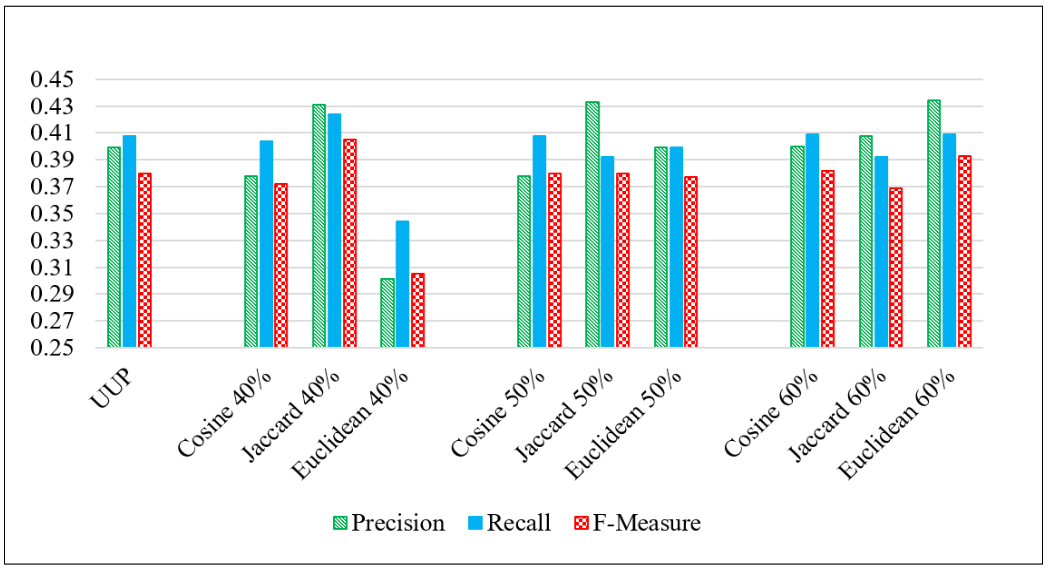


FIGURE 4.4: *Precision*, *Recall*, and *F-Measure* of QuPiD under Selected Similarity Methods and *mpeT* Values.

TABLE 4.4: Correctly Classified Queries by QuPiD Attack under Selected Similarity Methods and *mpeT* Values.

Similarity Method	***mpeT***	**Correctly Classified Queries**	**Correctly Classified Queries %**
Cosine Similarity	40%	4851	40.3947
	50%	5061	40.7521
	60%	5136	40.9145
Jaccard Similarity	40%	1772	42.4228
	50%	2427	39.1578
	60%	3315	39.2122
Euclidean Distance	40%	874	34.4366
	50%	1275	39.9311
	60%	1815	40.9152
UUP	-	5089	40.8132

data (i.e. similarity methods and *mpeT* values) in terms of *precision*, *recall*, and *f-measure*. According to the Fig. 4.4, the performance of the model in case of cosine similarity is almost similar to the performance of the PIR protocol without PEM in every scenario. While in the case of Jaccard similarity, the performance of the model is improved as compared to UUP scenario both in terms of precision and recall. However, in the case of Euclidean distance, the precision, and recall of the model decrease with the decrease in *mpeT* value. This decrease in precision and recall shows the PEM offers more privacy to the users with Euclidean distance even in case of machine learning attack. Table 4.4 and Fig. 4.5 shows the number of correctly classified queries by QuPiD attack under various *mpeT* values.

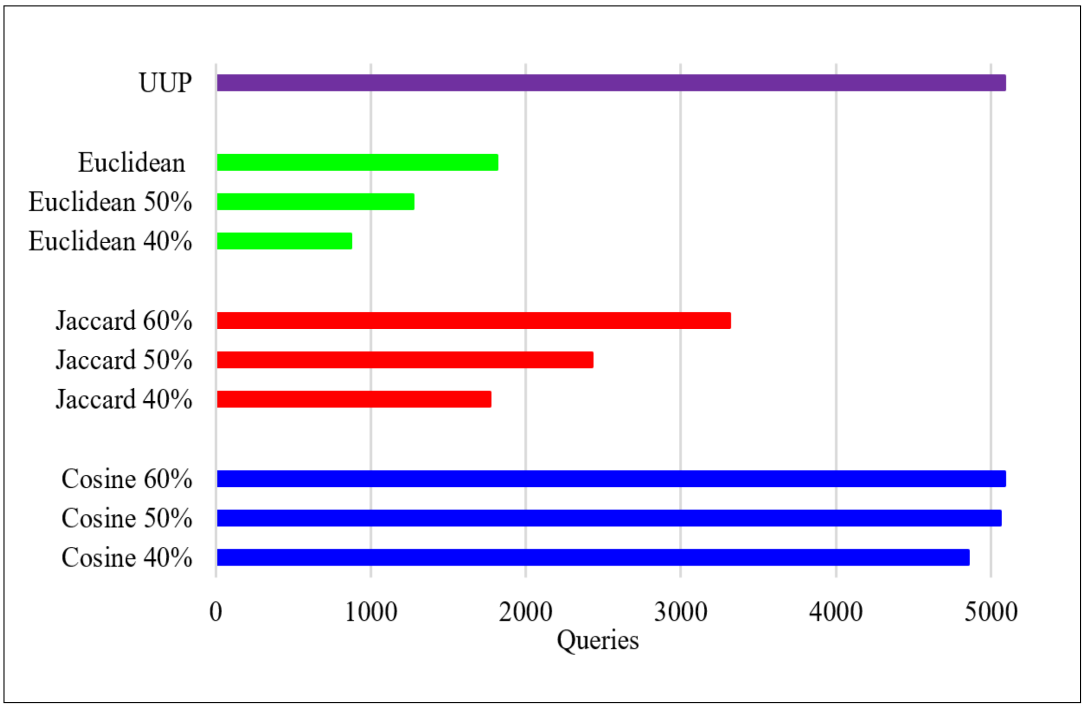


FIGURE 4.5: Correctly Classified Queries by QuPiD Attack under Selected Similarity Methods and $mpeT$ Values.

These results show that performance of the QuPiD Attack is significantly reducing in the presence of PEM as compered to UUP. We show that PEM clearly out performed UUP in case of QuPiD attack by reducing the attack precision and recall by 9.9% and 6.4% respectively.

4.3.3 Privacy Evaluation of PEM

In this section we discuss performance of Privacy Exposure Measure (PEM) and Useless User Profile (UUP)in terms of privacy metrics. We evaluate the privacy provided by PEM-powered-UUP and UUP in terms of Profile Similarity, Cross-Entropy loss and Kullback-Liebler (KL) divergence.

4.3.3.1 Profile Similarity

Profile similarity is one of the basic measure that calculate similarity between two profiles. In these experiments, we calculate the similarity between two profiles created through PEM and UUP. For similarity calculation we used Euclidean

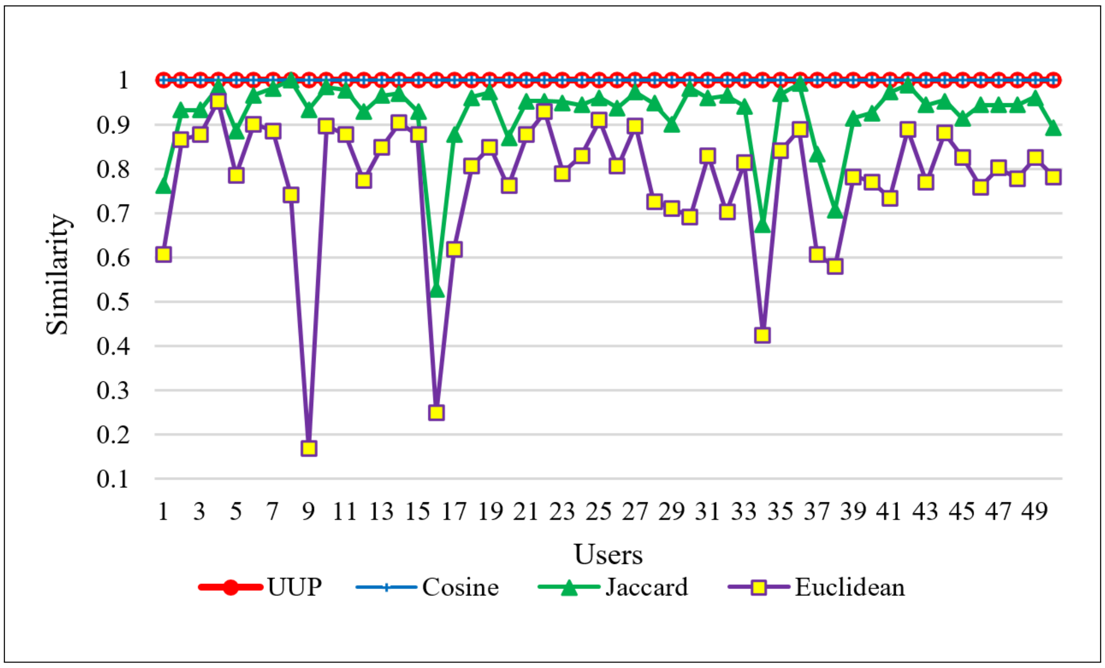


FIGURE 4.6: User Profile Similarity under Selected Similarity Methods with $mpeT = 60\%$.

distance (shown in Equation 4.5). The similarity is shown between "0" and "1" where "0" shows no similarity and "1" shows maximum similarity.

Fig. 4.6, Fig. 4.7, and Fig. 4.8 illustrate the similarity between selected users profiles created through PEM under $mpeT$ value of 60%, 50%, and 40% and UUP. Fig. 4.6 shows the user's profile similarity built with selected similarity methods when $mpeT$ is set to 60%. According to Fig. 4.6, the similarity between PEM and UUP profiles with cosine is 100%, which means 100% privacy exposure. However, the profile similarity in Jaccard similarity and Euclidean distance is less than 100% for every user.

Similarly, Fig. 4.7 depicts the user profile similarity with the $mpeT$ is set to 50%. Fig. 4.7 shows that except cosine, the profile created through Jaccard similarity and Euclidean distance is too much different as compared to UUP profile. In most cases, the profile built by Euclidean distance offers more privacy to the user as compared to cosine similarity and Jaccard similarity. While according to Fig. 4.8 when $mpeT$ is set to 40%, we can observe a significant distance between PEM and without PEM based user profile in all similarity methods. Moreover, Euclidean distance again out performed Jaccard similarity and cosine similarity methods.

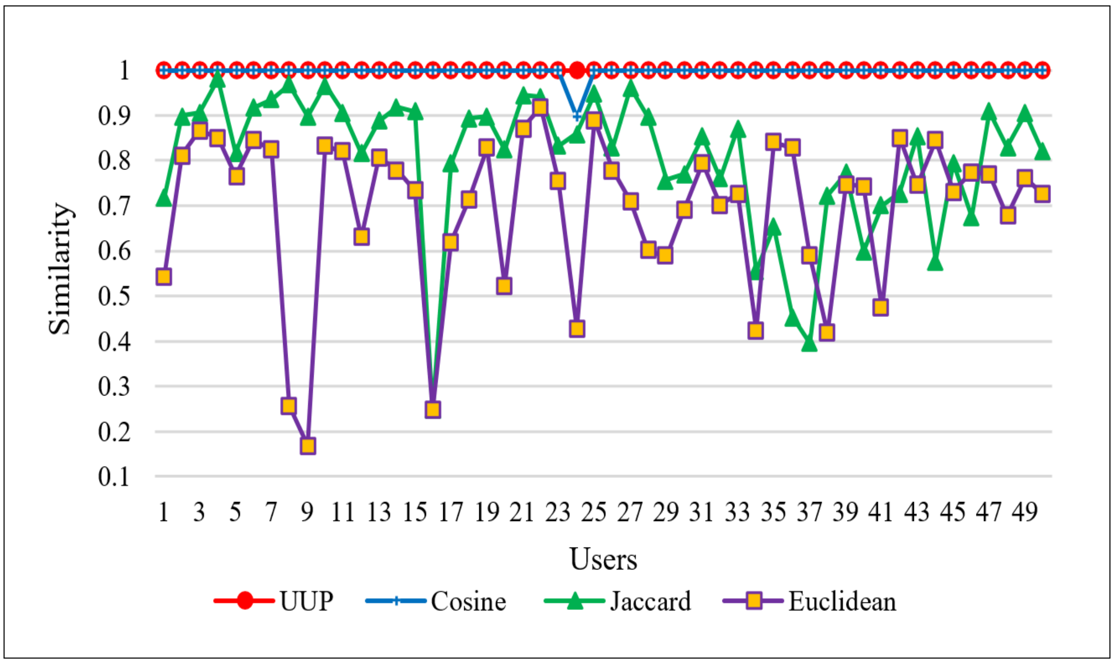


FIGURE 4.7: User Profile Similarity under Selected Similarity Methods with $mpeT = 50\%$. Where the Red Color line shows the Profile Similarity of UUP (Useless User Profile) Protocol. The Blue Color line shows the Profile Similarity of PEM with Cosine Similarity. The Green Color line shows the Profile Similarity of PEM with Jaccard Similarity. While The Purple Color line shows the Profile Similarity of PEM with Euclidean Distance.

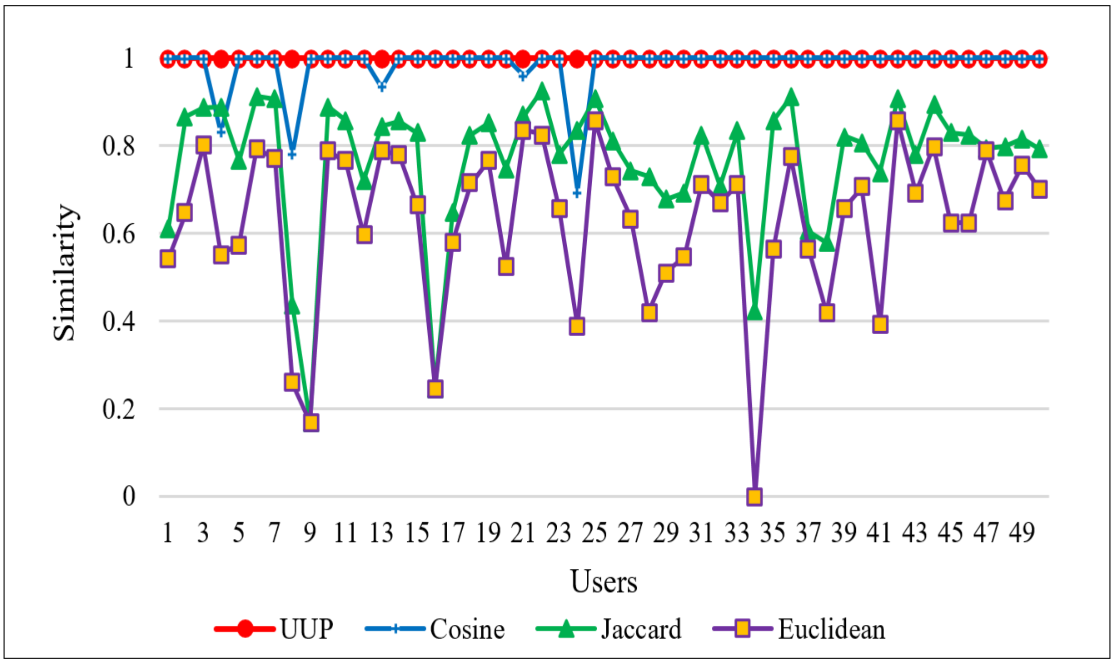


FIGURE 4.8: User Profile Similarity under Selected Similarity Methods with $mpeT = 40\%$. Where the Red Color line shows the Profile Similarity of UUP (Useless User Profile) Protocol. The Blue Color line shows the Profile Similarity of PEM with Cosine Similarity. The Green Color line shows the Profile Similarity of PEM with Jaccard Similarity. While The Purple Color line shows the Profile Similarity of PEM with Euclidean Distance.

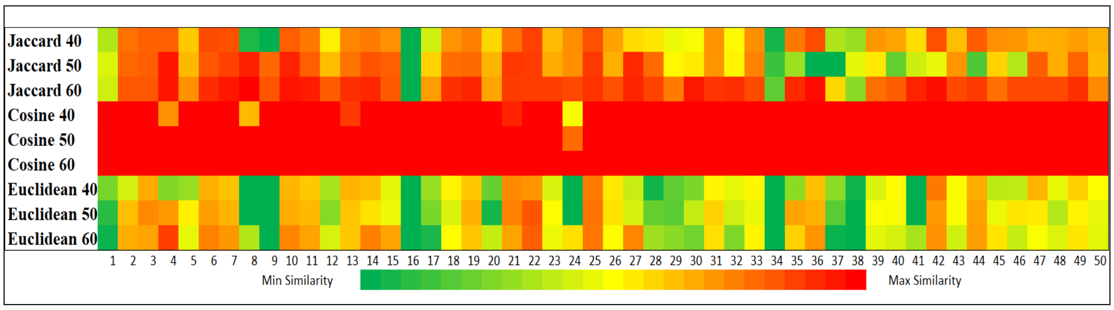


FIGURE 4.9: Pixel Map of Profile Similarity of all Users.

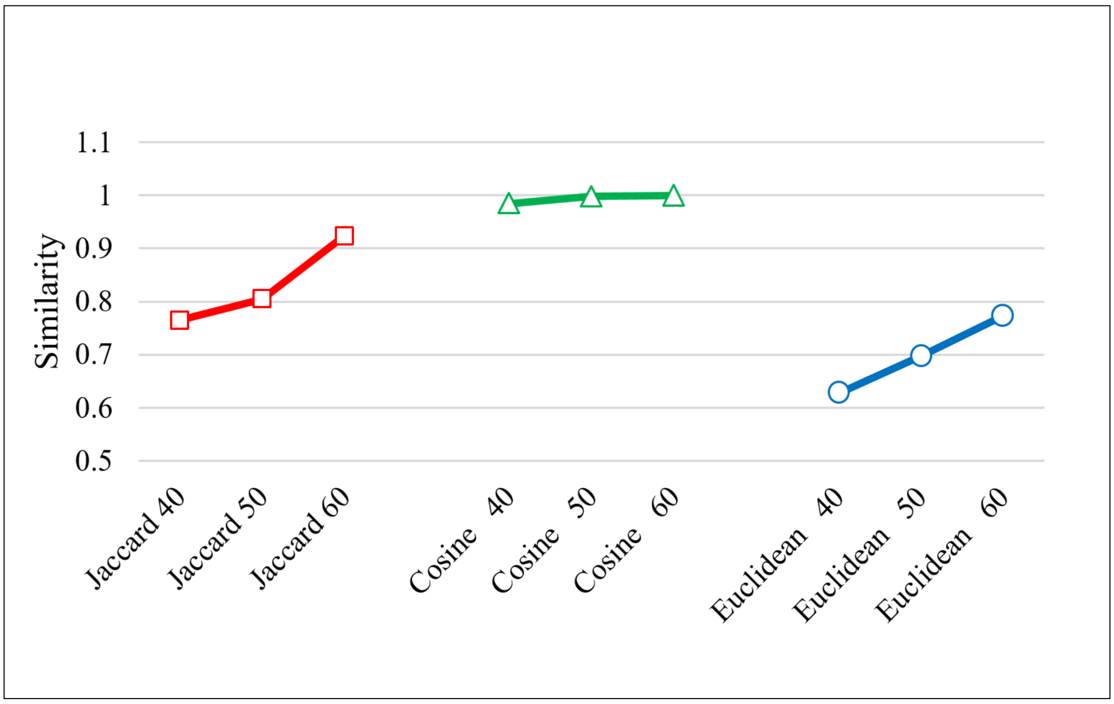


FIGURE 4.10: Average Profile Similarity.

In the overall performance, the average profile similarity between Privacy Exposure Measure (PEM) and UUP protocol scenario in the case of cosine similarity is 98.4%, 99.8%, and 100% for the $mpeT$ value of 40%, 50%, and 60%, respectively. Similarly, in the case of Jaccard similarity, the average profile similarity is 76.5%, 80.5%, and 92.4% for the $mpeT$ value of 40%, 50%, and 60%, respectively. In the case of Euclidean distance, the average profile similarity recorded as 62.8%, 69.7%, and 77.4% for the $mpeT$ value of 40%, 50%, and 60%, respectively. Fig. 4.9 shows the pixel diagram of profile similarity between the profiles made with and without PEM. The pixel in dark green color shows the minimum similarity while the pixel in red color shows maximum similarity. Similarly, the average profile similarity in each scenario is shown in Fig. 4.10.

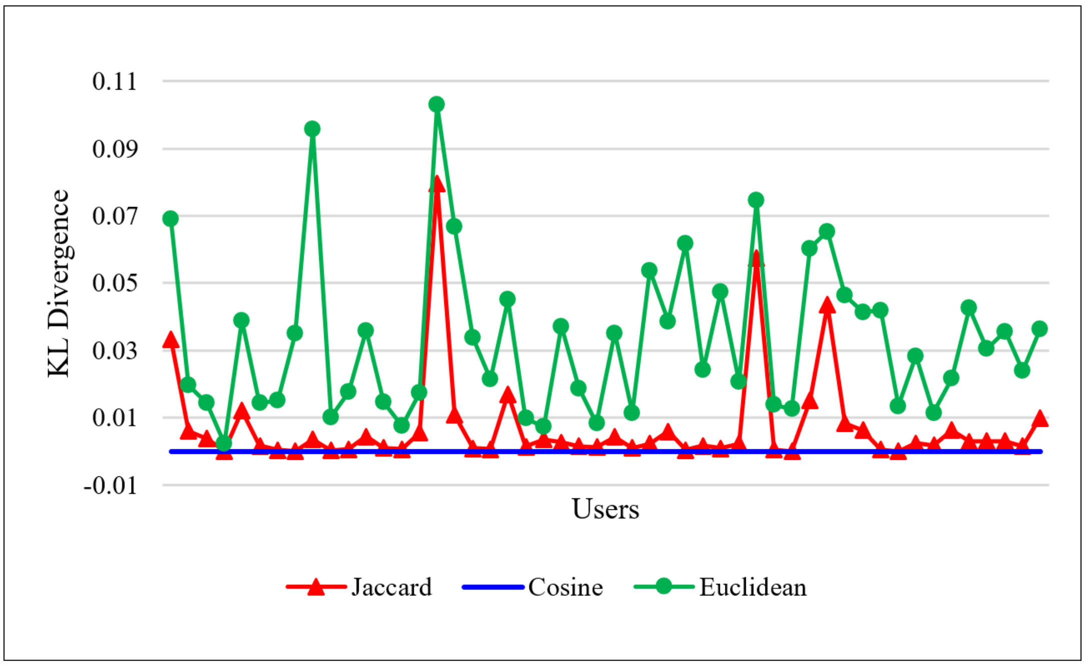


FIGURE 4.11: User Profile KL Divergence under Selected Similarity Methods with $mpeT$ = 60%.

4.3.3.2 Kullback-Liebler (KL) Divergence

This section contains discussion about Kullback-Liebler (KL) Divergence between user profiles created through PEM and UUP. KL Divergence is also known as relative Entropy that is used to measure the divergence between one probability distribution with other reference probability distribution. In these experiments, reference probability distribution is the user's profile created through UUP.

Fig. 4.11, Fig. 4.12, and Fig. 4.13 shows the KL Divergence between User's UUP profile and PEM profile with selected $mpeT$ values. Figure 4.11, depicts the KL Divergence between user's UUP profile and PEM profile when $mpeT$ value is set to 60%. The results showed that the profiles created through Euclidean distance of almost every user is more diverged as compere to Jaccard and cosine. The profiles created through cosine is almost same to the reference profiles (i-e UUP profiles). While the profiles created through Jaccard is slightly diverged from the reference profiles of the users.

Similarly in case of $mpeT$ value of 50% and 40%, the users' profiles created through cosine and Jaccard are less diverged from the reference profiles as compere to the

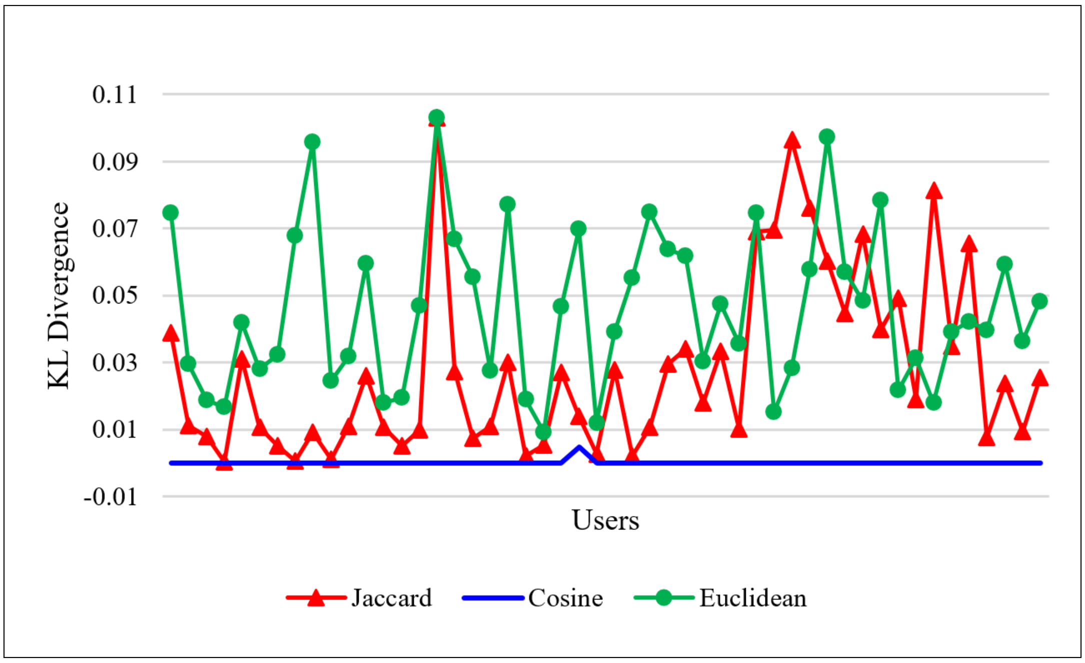


FIGURE 4.12: User profile Kullback-Liebler (KL) Divergence under Selected Similarity Methods with $mpeT = 50\%$. Where the Red Color line shows the Kullback-Liebler (KL) Divergence of User Profile with Jaccard Similarity. The Blue Color line shows the Kullback-Liebler (KL) Divergence of User Profile with Cosine Similarity. While the Green Color line shows the Kullback-Liebler (KL) Divergence of User Profile with Euclidean Distance.

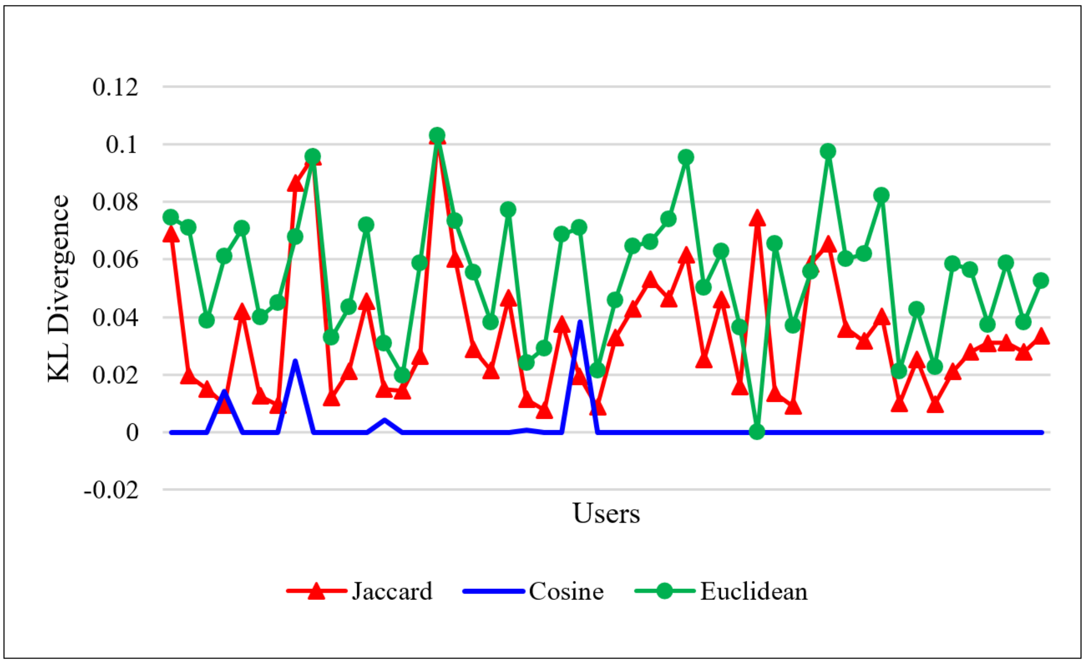


FIGURE 4.13: User Profile Kullback-Liebler (KL) Divergence under Selected Similarity Methods with $mpeT = 40\%$. Where the Red Color line shows the Kullback-Liebler (KL) Divergence of User Profile with Jaccard Similarity. The Blue Color line shows the Kullback-Liebler (KL) Divergence of User Profile with Cosine Similarity. While the Green Color line shows the Kullback-Liebler (KL) Divergence of User Profile with Euclidean Distance.

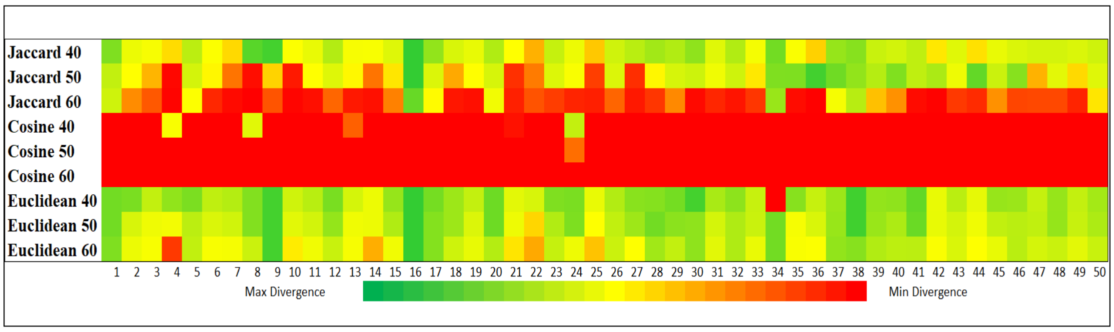


FIGURE 4.14: Pixel Map of KL Divergence between Profiles of all Users.

profiles created through Euclidean distance (as shown in Fig. 4.12 and Fig. 4.13).

Similarly, Fig. 4.14 shows the pixel diagram of KL Divergence. Each pixel shows the KL Divergence between the profiles made with and without PEM. The pixel in dark green color shows the maximum divergence while the pixel in red color shows minimum divergence. Maximum KL Divergence shows the maximum distance between profiles or maximum user privacy. The results show that for the $mpeT$ value of 40%, the KL Divergence between the user's profiles is maximum with Euclidean distance. This shows that the maximum user's privacy in Euclidean distance with the $mpeT$ is 40%. Similarly, for the $mpeT$ value of 50% and 60% with the Euclidean distance, the value of KL Divergence is greater than Jaccard and Cosine similarity. The average profile KL Divergence in each scenario for each user is shown in Fig. 4.15. Form the results, it is concluded that PEM with Euclidean distance and lower $mpeT$ value offers better privacy to the user.

4.3.3.3 Cross-Entropy Loss

This section contains discussion about Cross Entropy Loss between user profiles created through PEM and UUP. Usually Cross-Entropy loss is used to measure the difference between two probability distributions. However, in the case of privacy exposure, Cross-Entropy loss is used to determine the difference or loss between the user's actual profile and profile created through any privacy protection mechanism. In this research, Cross Entropy Loss is used to measure the difference or loss between user's actual profile (created through Privacy Exposure Measure (PEM)) and profile created through Useless User Profile (UUP) protocol.

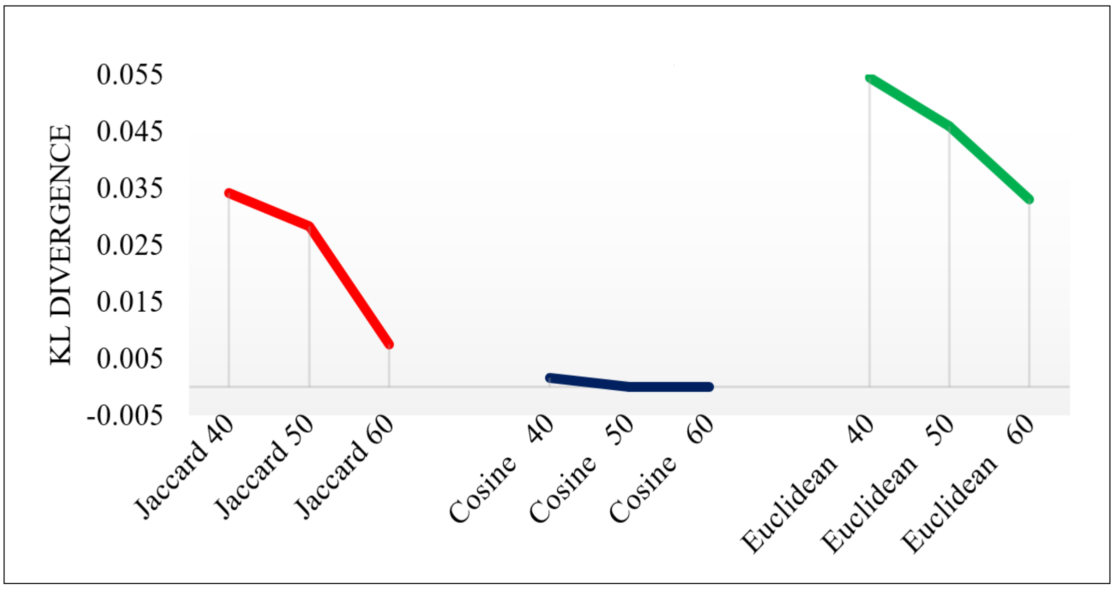


FIGURE 4.15: Average Profile KL Divergence.

Fig. 4.16, Fig. 4.17, and Fig. 4.18 shows Cross-Entropy loss between User's UUP profile and PEM profile with selected $mpeT$ values. Fig. 4.16, depicts the Cross-Entropy loss between user's UUP profile and PEM profile when $mpeT$ value is set to 60%. The results showed that the profiles created through Euclidean distance is have more information loss as to Jaccard and Cosine. The profiles created through Cosine is almost same to the reference profiles (i-e UUP profiles). While the profiles created through Jaccard have low information loss.

Similarly in case of $mpeT$ value of 50% and 40%, the users' profiles created through cosine and Jaccard have less information loss from the reference profiles as compere to the profiles created through Euclidean distance (as shown in Fig. 4.17 and Fig. 4.18).

In a nutshell, it can be concluded that for the $mpeT$ value of 40%, the Cross-Entropy loss of the users' profiles is maximum with Euclidean distance. This shows that the maximum user's privacy in Euclidean distance with the $mpeT$ is 40%. Similarly, for the $mpeT$ value of 50% and 60% with the Euclidean distance, the value of information loss is greater than Jaccard and Cosine similarity. The average Cross-Entropy loss in each scenario for each user is shown in Fig. 4.19. Form the results, it is concluded that PEM with Euclidean distance and lower $mpeT$ value offers better privacy to the user as compere to UUP.

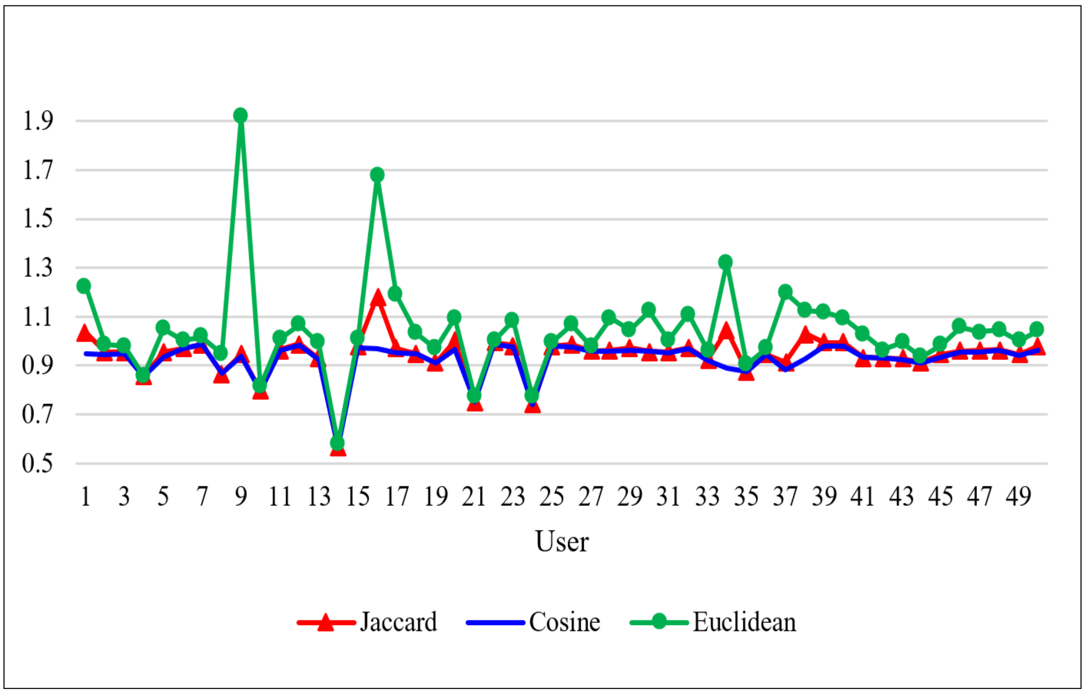


FIGURE 4.16: User Profile Cross-Entropy Loss under selected Similarity Methods with $mpeT = 60\%$. Where the Red Color line shows the Cross-Entropy Loss of User Profile with Jaccard Similarity. The Blue Color line shows the Cross-Entropy Loss of User Profile with Cosine Similarity. While the Green Color line shows the Cross-Entropy Loss of Users profiles with Euclidean Distance.

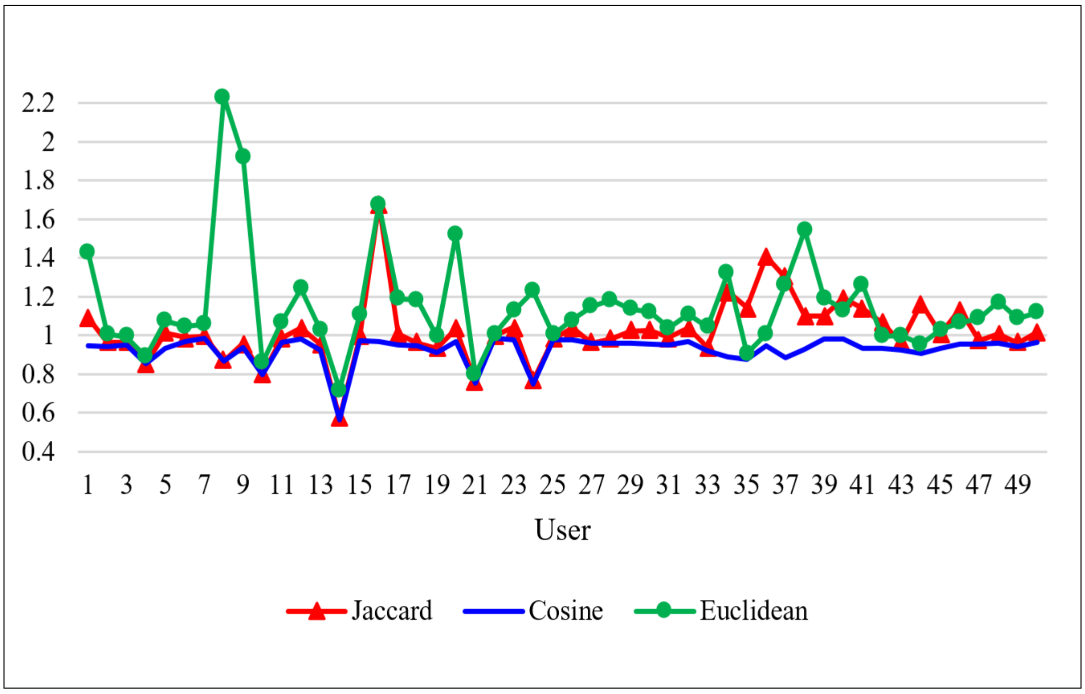


FIGURE 4.17: User Profile Cross-Entropy Loss under Selected Similarity Methods with $mpeT = 50\%$. Where the Red Color line shows the Cross-Entropy Loss of User Profile with Jaccard Similarity. The Blue Color line shows the Cross-Entropy Loss of User Profile with Cosine Similarity. While the Green Color line shows the Cross-Entropy Loss of Users profiles with Euclidean Distance.

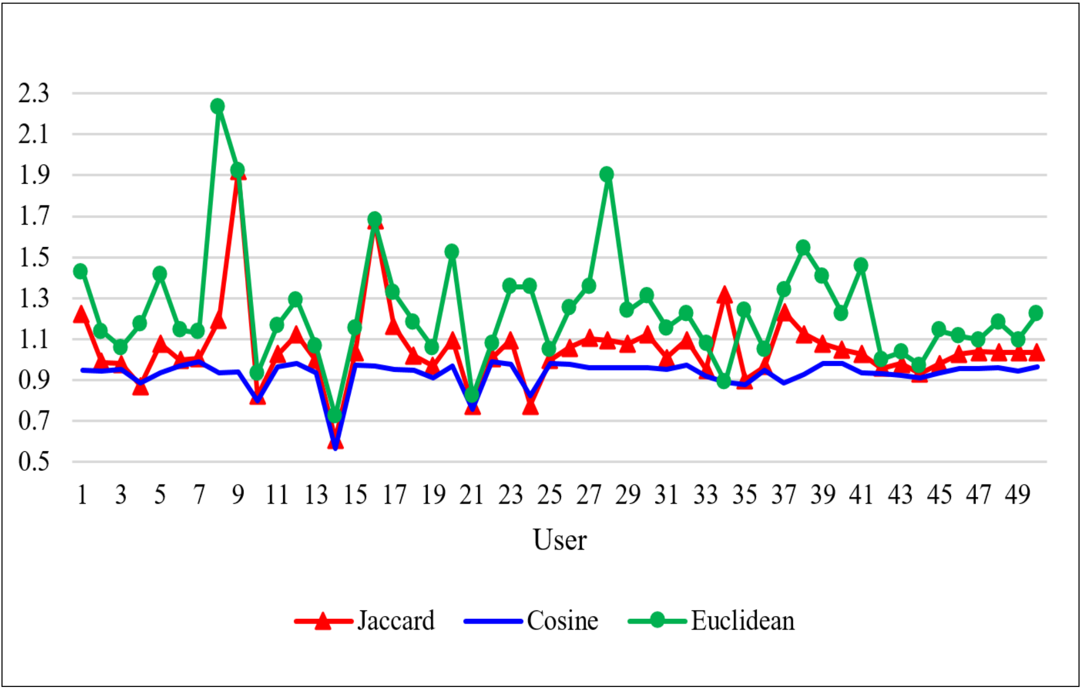


FIGURE 4.18: User Profile Cross-Entropy Loss under Selected Similarity Methods with $mpeT = 40\%$. Where the Red Color line shows the Cross-Entropy Loss of User Profile with Jaccard Similarity. The Blue Color line shows the Cross-Entropy Loss of User Profile with Cosine Similarity. While the Green Color line shows the Cross-Entropy Loss of Users profiles with Euclidean Distance.

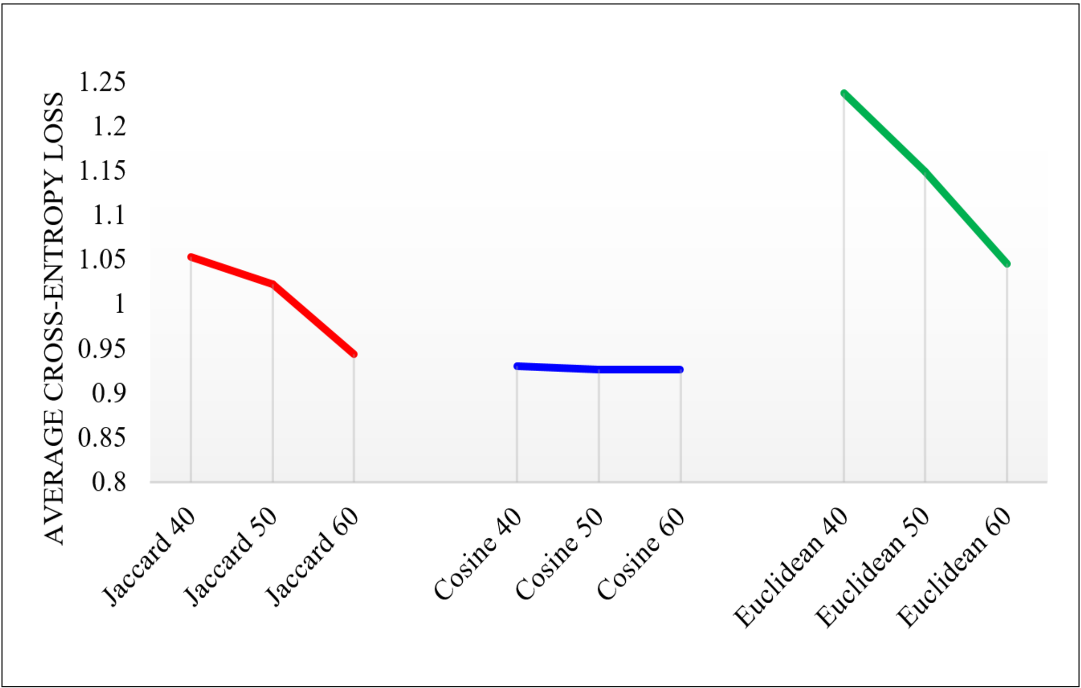


FIGURE 4.19: User Profile Average Cross-Entropy Loss under Selected Similarity Methods. Where the Red Color line shows the Cross-Entropy Loss of User Profile with Jaccard Similarity. The Blue Color line shows the Cross-Entropy Loss of User Profile with Cosine Similarity. While the Green Color line shows the Cross-Entropy Loss of Users profiles with Euclidean Distance.

4.3.4 Performance of PEM with Various *mpeT* Values

In Privacy Exposure Measure (PEM), *mpeT* value plays an important role to ensure user's privacy as it is used by the users to set their maximum privacy exposure to the adversarial web search engine. In the previous section, we discussed the results of the *mpeT* value of 40%, 50%, and 60% which yields better results in terms of privacy provided to the user. However, the *mpeT* value of 40%, 50%, and 60% gives us the average picture of the user's privacy exposure in the case of Privacy Exposure Measure (PEM). As the user has the luxury to set any *mpeT* value between 0% and 100% therefore, to investigate the users' privacy exposure in the case of Privacy Exposure Measure (PEM), we conducted the experiments of Privacy Exposure Measure (PEM) with *mpeT* values between 10% and 100% and Euclidean Distance. Fig. 4.20 shows the number of permissible queries by Privacy Exposure Measure (PEM) with *mpeT* values between 10% and 100%. While Fig. 4.21 depicts the profile similarity of all users when *mpeT* values between 10% and 100% are used.

The results show that the number of permissible queries is very low (4.03%) when *mpeT* value is set to 10% which means maximum privacy. However, with the increase of *mpeT* value, the number of permissible queries increases and the risk of privacy exposure is high. Similarly, Fig. 4.21 depicts the profile similarity of all users when *mpeT* values between 10% and 100% are used with their non-PEM profile. The pixel in dark green color shows the minimum similarity while the pixel in red color shows maximum similarity. According to the Fig. 4.21, most of the pixels of *mpeT* value 10%, 20%, 30%, and 40% are in dark green color which shows that profiles created with these *mpeT* values have minimum similarity with the profiles of the same users created when Privacy Exposure Measure (PEM) is not used. While for *mpeT* value of 80% and 90%, almost every pixel is red or orange which shows maximum similarity or maximum privacy exposure. Although it is a tough question for the user to decide the privacy exposure value as it also prevents the user from submitting desired queries due to high exposure. However, this is the price user have to pay in order to get high privacy while using PEM.

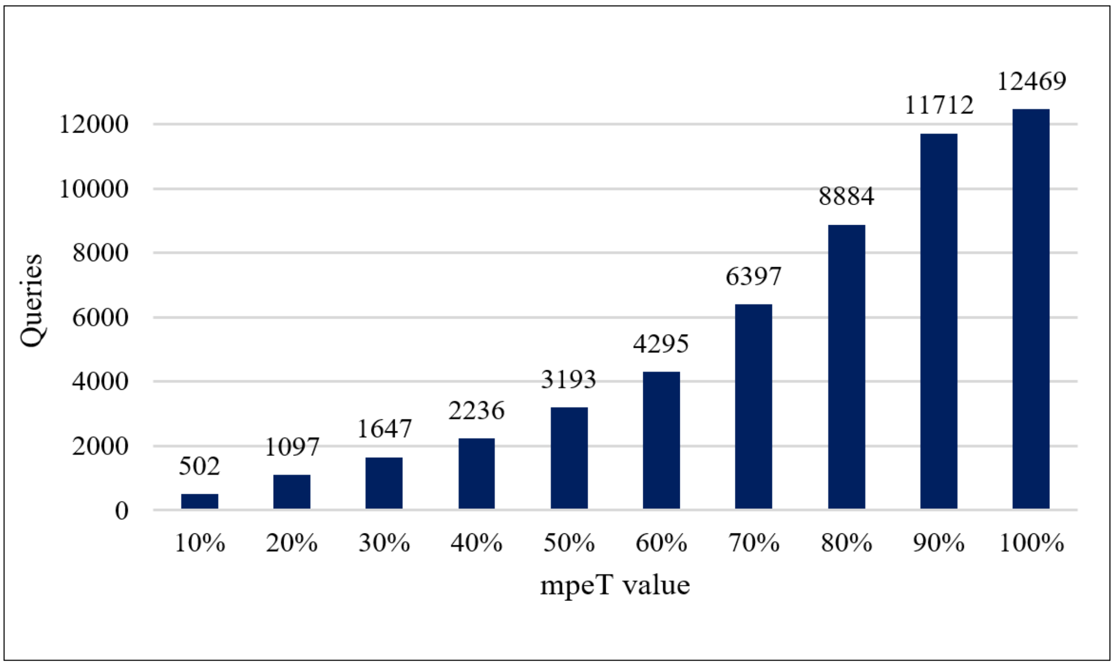


FIGURE 4.20: Permissible Queries by PEM with *mpeT* Values between 10% and 100%.

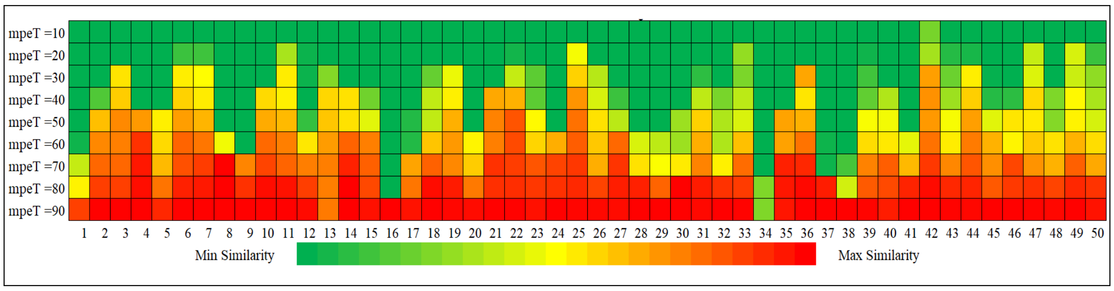


FIGURE 4.21: Profile similarity of all users when *mpeT* Values between 10% and 90%.

4.3.5 Trade-off Between Privacy and Retrieved Results

Privacy is a sensitive issue from the user's point of view. Unfortunately for web search engines, user privacy is just a behavior tracking and used for multiple purposes including profit. However, unfortunately, profiling user behavior is also important for the quality of the results retrieved by the web search engine. According to the working of the web search algorithm the retrieval and ranking of query results depend upon the user's history [152]. Web search engine uses users' history to understand their preferences automatically. Therefore in order to get the most relevant results against a query, a proper user profile is needed that inevitably can breach user's privacy as well.

In the case of PEM, users can set their profile exposure level in order to limit their privacy exposure. However, PEM provides privacy to the user by limiting the number of queries. According to the working of PEM, if a query's exposure level is greater the $mpeT$ value then it is reverted to the user for reconsideration. Users can either change the query or can stay with the same query. It is a tough question for the user to decide the privacy exposure value as it also prevents the user from submitting desired queries due to high exposure.

The lower $mpeT$ value can provide higher privacy to the user but it can affect the quality of the retrieved results. As PEM put a limit on the number of queries, therefore it might also affect the result ranking process. Consequently, the user will have to think about every query in order to avoid the privacy exposure threshold which may ultimately put users in an annoyed state. Refer to Fig. 4.20, if $mpeT$ value is set to 20% or 30% then only 8.8% or 13.21% queries will be forwarded freely. While the user will have to think about the rest of the 91.2% or 86.79% queries and will also get a lower quality of results. However, this is the drawback of this technique which user have to endure in order to get high privacy.

4.3.6 Limitations of PEM

In the previous chapter, we showed that our proposed QuPiD attack can successfully break the privacy provided by the PIR protocols. While in order to improve the effectiveness of the PIR protocol in terms of privacy and against the QuPiD attack, we proposed PEM in this chapter. PEM improved the privacy provided by the PIR protocols effectively. However, PEM has some limitations as well. These limitations are discussed as follow:

1. One of the major limitations of the PEM is the quality of the retrieved results. As the web search engine uses the user's history to refine the result and due to PEM the history of the user may be imperfect.
2. It may put the user in an annoying state if the user is asked each time to reconsider his/her query due to higher exposure.

3. It may also affect the ranking process of the result due to limited web search history.

4. The user profile is created and tested based on 10 major classes of interest. This is one of the vital limitations of PEM as user's interests cannot be restricted to ten topics.

4.4 Conclusion

In this chapter, we present Privacy Exposure Measure (PEM), a privacy exposure minimization technique for the private information retrieval protocols against QuPiD attack. Privacy Exposure Measure (PEM) is used to estimate the user's privacy exposure by calculating the similarity between a user's profile and query. Moreover, a user can also set his/her maximum privacy exposure threshold ($mpeT$) value.

To evaluate the performance of PEM in terms of privacy, we conducted experiments with three similarity methods (i.e., cosine, Jaccard, Euclidean) under the $mpeT$ value of 60%, 50%, and 40%, respectively. For performance evaluation, we used, QuPiD attack, profile similarity, Kullback-Liebler (KL) Divergence, and Cross-Entropy loss. The results showed that Euclidean distance permits very few queries with all three $mpeT$ values as compare to Jaccard and cosine. Upon data level analysis it was revealed that our data is composed of vector quantities and Euclidean distance use component to component matching. Due to this component to component matching, Euclidean distance is more sensitive as compere to Cosine and Jaccard similarity. Regarding the success of QuPiD attack, PEM performed better than UUP, as the precision and recall of QuPiD attack in case of PEM was 9.9% and 6.4% less respectively then UUP. Overall, the success rate (f-measure) of QuPiD attack recorded between 30.5% and 40.5% in different scenarios. The performance of PEM is also evaluated for $mpeT$ values between 10% and 100% and Euclidean Distance and it was concluded that low $mpeT$ value provides higher privacy to the user but it can affect the quality of the retrieved results.

Similarly, in case of privacy evaluation the performance of PEM (Privacy Exposure Measure) and UUP (Useless User Profile) is evaluation in three measures i-e. Profile Similarity, KL Divergence, and Cross-Entropy loss. We show that the users' profiles created through PEM are different and diverged as compered to the profiles created through UUP (Useless User Profile). Moreover the profile created through euclidean distance has much information loss as compered to profile created through UUP (Useless User Profile). Therefore, from the results it is concluded PEM (Privacy Exposure Measure) offer more privacy to the user as compered to UUP (Useless User Profile) and can effectively preserve user privacy.

Chapter 5

Conclusion and Perspectives for Future Work

5.1 Conclusion

Controlling private data is becoming increasingly important in today's world. The technological advancements in the field of information technology enabled online service providers to easily process the huge amount of data. These online service providers collect and analyze the information from billions of users. Although they claimed that the collection is used to improve service and user experience. However, online service providers mainly use the collected information for target advertisement and track personal data for insurance companies and similar businesses. This is especially a problem with web search services where billions of users submit queries to find their desired information on the web on a daily basis. These queries may contain privacy-sensitive information (such as health, investment, etc.) or information that can be used to infer user's age, gender, health condition, personal interests, religious or political affiliation, etc. This indiscriminate collection of users' information may cause critical privacy breach. To address the issue of privacy infringement, researchers have proposed several techniques and alternatives that help users to control their privacy exposure. These techniques can be classified into four major classes, i.e., user anonymizing networks, profile obfuscation, private information retrieval (PIR) protocols, and hybrid techniques. Similarly, there are several studies and privacy attacks available that indicate that these techniques and solutions cannot offer satisfactory privacy to the user. In this thesis, we studied such type of techniques, deficiencies, and attacks. More precisely, the focus of this thesis is to study the weakness and the level of protection offered by the Private Information Retrieval (PIR) protocols. To study the effectiveness of the Private Information Retrieval (PIR) protocols, we assumed that a web search engine is an adverse entity that is interested in gathering user information (queries). Furthermore, it is also assumed that the web search engine has the users' previous (non-protected) queries i.e., before the users employ any privacy preservation solution.

5.1.1 Robustness of Existing PIR Protocols

In the third chapter of the thesis, we have evaluated the performance of the available Private Information Retrieval (PIR) protocols in terms of privacy. In the literature, there are some attacks available for the performance evaluation of privacy preserve web search or private web search. However, these attacks are proposed to evaluate the performance of indistinguishability and unlinkability solutions. Moreover, these attacks cannot be used for the evaluation of Private Information Retrieval (PIR) protocols as they are classification model was bi-class. Therefore, we proposed a QuPiD attack: a machine learning based attack that evaluates the effectiveness of PIR protocols in privacy protection. The primary aim of QuPiD attack is to associate the query to the correct user (in case of PIR protocol). QuPiD Attack determines the distance between the user's Profile (web search history) and upcoming query using a novel feature vector. The QuPiD attack succeeds in associating a larger proportion of queries to the correct user in a minimum amount of time. For instance, the most successful classifier (IBk) of the QuPiD attack took 15.13 seconds to classify associate the queries of 100 users' dataset with the accuracy of 45.1%. In addition to that, unlike other privacy evaluation models that only used user queries strings for training, we used queries strings and topic score feature vector which improve the accuracy of the attack.

Furthermore, we evaluated the performance of PIR protocol from different aspects (using QuPiD attack) including the impact of group size, training data size, and number queries submitted in a session by the user over the privacy exposure. The results show that a bigger group size offers more protection to the user. However, PIR protocols cannot offer bigger group size due to latency in practical. Therefore, even with the group size of 10 users, QuPiD attack successfully able to associate 48.83% queries to the correct user. Similarly, in the next experiment, we evaluated the effect of a number of queries submitted by the user in a session over a user's privacy disclosure. The results showed that the accuracy of QuPiD attack is almost the same in the revelation of user queries. To evaluate the impact of training data size, we conducted experiments with 25%, 50%, 75%, and 100% profile size of the available data. It was found that the increase in profile has a mild effect on the disclosure of the user's privacy. However, this experiment confirms the obvious fact that a bigger user profile size is more prone to maximum privacy disclosure. In the last experiment, we found that PIR protocols are more prone to the QuPiD attack between 1:00 am and 9:00 am due to low traffic.

We confirmed by evaluating the PIR protocol using a dataset of real queries released by AOL from different angles that QuPiD attack can effectively infringe the user's privacy. We, therefore, concluded in a satisfactory manner that PIR protocol stood weak against QuPiD attack and cannot provide adequate privacy to the user.

5.1.2 Topics Score: A Potential Addition in Feature Vector Inventory

In machine learning, features play an important role in the identification of a certain entity or event. The feature vector is a set of different features (i.e., numeric, text, pixels and others depend on the domain of the problem), representing an object or an event. As most of the machine learning algorithms are based on statistical formulae, therefore, they need most of the problems to be presented in a numeric form. Despite the previous attempts, we proposed Topics Score feature vector for the classification model. In previous works, all attributes available in AOL dataset were used to build the classification model. However, they mostly depend on the query strings and the decisions were made based on string similarity which cannot compute semantic similarity between strings due to which the performance of the attack affected. Therefore, we proposed a Topics Score feature vector to improve the effectiveness of the attack. We conducted experiments to evaluate the performance of the classification model trained with (i) Strings (Queries), (ii) Topic Score, and (iii) Strings + Topic Score over datasets of 100, 200, 300, and 500 users. The results showed that the performance of the Topic Score feature vector is 2.83% performed better as compared to the strings feature vector on the average. However, if we use both Topic Score and Strings as feature vector the performance of the attack is increased by 6.93% on the average. This increase in the performance shows that the Topic Score is a potential candidate to be used as a feature vector in the future. Moreover, the Topic Score feature vector also enabled QuPiD attack to use semantics in the similarity calculation process and thus increases the capability of the attack.

5.1.3 Session Window

Another major contribution of this work is estimating the appropriate size of the session window for QuPiD attack. Session window is a block of records in which query/queries associated with the target user are present but might be against another user (depending on the protocol). In other words, the session window is composed on the selected number of entries (or records) in the web search engine query log, appeared immediately before and after the query of UoI or QoI. Session window is used to reduce the number of records to be tested for associating with queries to the correct user. We have proposed time-based and record (query entries) based session windows for different situations.

5.1.4 A Privacy Exposure Minimizing Technique for Private Information Retrieval Protocols

Since the PIR protocol cannot offer adequate privacy to the user, we designed the PEM, a privacy exposure estimation module for private information retrieval protocols to reduce the user's privacy exposure. PEM assesses the similarity between the user's profile and the query before posting to the shuffling process of the PIR protocol and assists the user in avoiding privacy exposure. The advantage of the proposed approach is that the user can avoid QuPiD attack by restricting the queries with higher profile exposure. Since the web search engine already has the user's real profile (history); we used a reactive approach in the PEM by allowing those queries whose similarity is less than the threshold value. In this way, the success rate of user identification remains low in case of QuPiD attack.

We empirically evaluate the performance of the PEM-powered-PIR protocol using AOL data. The privacy evaluation of PEM-powered-PIR protocol is conducted using QuPiD attack and furthermore evaluated using profile similarity and Kullback-Liebler (KL) divergence. PEM-powered-PIR protocol is tested under three maximum privacy exposure Threshold ($mpeT$) values i.e. 60%, 50% and 40% using three similarity methods i.e. Cosine, Jaccard, and Euclidean. The results showed that Euclidean similarity permits very few queries with all three $mpeT$ values as compare to Jaccard and Cosine. The success rate of QuPiD attack was between 33% and 41% in different scenarios. However, due to low number of permissible queries in some scenarios maximum profile distance and the KL divergence proved that PEM-powered-PIR protocol can effectively preserve user privacy. Moreover, it can offer ample privacy to the user even in the case of QuPiD attack.

Although, Privacy Exposure Measure (PEM) improved the privacy provided by the Private Information Retrieval (PIR) protocols effectively. However, PEM has some limitations as well. One of the major limitations of the PEM is the quality of the retrieved results. As the web search engine uses the user's history to refine the result and due to PEM the history of the user may be imperfect. It may also affect the ranking process of the result due to limited web search history.

5.2 Perspectives for Future Work

Privacy of online user is a delicate problem and now taken as an important issue in every field especially after the emergence of web search engines and social media. As the web search engines collect user's personal information for many legitimate and illegitimate reasons such as result ranking and profit. Therefore, many techniques have been proposed that ensure user privacy while using web search engines. This thesis presents the performance evaluation of prominent private information retrieval protocols in terms of privacy. In this section, we present some perspective for future work. First, we discuss the settings of this thesis i.e., our proposed adverse model (QuPiD attack) for privacy evaluation of PIR protocols. Then we give some perspectives regarding the future work of the proposed privacy evaluation model (QuPiD attack) and suggested modification (PEM) in Private Information Retrieval (PIR) protocols.

5.2.1 Adverse Model

In order to test that how much privacy Private Information Retrieval (PIR) protocols can provide to the user, we proposed a Machine-Learning attack (QuPiD attack) and consider a scenario. In this scenario the Web search engine is considered as adverse entity that is interested in finding the original queries of those users who use Private Information Retrieval (PIR) protocols to ensure their privacy. We assume that the web search engine will launch the Machine-Learning Attack (QuPiD Attack) based on the users' history. For testing we used the session window to capture the transactions of other group members. However, in a more realistic scenario, it is very difficult to identify the transaction of the whole group due to the bigger session window size. Therefore, capturing, and cleaning of the session window with a group transaction is an open challenge in future research. The similar case of a new user whose profile (history) is not yet available to the search engine, presents another challenge for the future. Moreover, in future works, we will investigate how web search engines maintain users' profiles currently.

5.2.2 QuPiD Attack

We have seen in this thesis that QuPiD attack used different machine learning algorithms to build the classification model. Some of these algorithms such as IBK, Bagging, and J48 showed promising results. Due to the feature vector used, the performance of these algorithms has improved. In contrast to the existing attacks (Peddinti and Saxena [14, 45, 48], Petit et al. [2, 46] and Gervais et al. [111]), QuPiD attack used a query string and "Topic Score" as the feature vector. Apart from the selected five best classification algorithms, we also tested the performance of the QuPiD attack with Recurrent Neural Network (RNN) using Long Short-Term Memory (LSTM) which provide betters results with textual data. However, we believe that the performance of such kind of attacks can be further improved using other more sophisticated machine learning techniques such as Artificial Neural Network with fine-tuned parameters and functions, and techniques of Natural Language Processing (NLP).

Another limitation of the privacy attacks is the lack of assessing semantic relationship existing in the queries. The privacy attacks presented in the literature usually rely on string matching techniques. However, the string matching techniques cannot be used to compute semantic relatedness. For instance, queries like "cricket" and "football" belong to the same topic, i.e., "Sports", however, string matching cannot compute relatedness between them. The strength of the QuPiD attack is the Topics Score feature vector, which provides a query score in ten major topics including Sports, Society, Science, Recreation, Home, Health, Games, Computers, Business, and Arts. Using the Topics Score, QuPiD attack can find semantic relatedness between the queries and user profile. Although since the users' interests cannot be restricted to ten topics, a better similarity algorithms might be useful. Future work should focus on a better similarity calculation algorithm for the Topic Score.

During the experimentation, we discovered an unexpected unsteady behavior of the QuPiD attack that was captured in the *recall* rate of the attack. We referred to this behavior as the ProQSim (Profile to Query Similarity) Effect. Upon investigation, it was found that this behavior of the QuPiD attack occurred due to the following reasons.

1. Firstly, the lack of traces of the user's interests in the incoming query in the user profile.
2. Secondly, the presence of more than one almost similar user profile.

These reasons show that the user profile is not rich enough to cover all his/her interests either due to the small period of data (i.e., 3 months) and/or the limited categories in Topic Score feature vector. Therefore, a better similarity calculation algorithm might be useful to avoid the ProQSim effect.

5.2.3 Need for a Recent Benchmark Dataset

For the performance evaluation of the PIR protocols in the presence of QuPiD attack, we used the AOL dataset that was released in 2006. Nevertheless, since from the birth, Web is changing rapidly with the emergence of new technologies (e.g., jQuery, Underscore, CSS3, etc.), new services (e.g., social networks, etc.), and Smartphone. These evaluations in technology have greatly influenced the users' online behavior. In 2006, Google processed less 350 Billion queries while in 2012 they processed about 1.2 Trillion queries. Therefore, it is certain that the AOL dataset does not reflect the accurate picture of today's web search engine log. Moreover, the AOL dataset only contains 3 months of user queries, which cannot encompass the major portion of users' interests. Similarly, most of the users in the dataset issued very few queries in the entire period. This behavior is not very common for current online users. Apart from these major deficiencies in the dataset, we are forced to use this dataset for experimentation due to lack of available data. Other evaluation mechanisms and attacks in the literature also used the same dataset such as Peddinti and Saxena [14, 45, 48], Petit et al. [2, 46], and Gervais et al. [111]. It is quite a mystery that how web search engines are currently maintained users' profiles as they do not publish them.

5.2.4 PEM: Suggested Modification in Private Information Retrieval Protocols

In this thesis, we established that the ability of QuPiD attack in PIR protocols can be effectively minimized using Privacy Exposure Measure (PEM). The user can set his or her maximum privacy exposure while querying the web search using the PIR protocol. The advantage of our approach is that the user can avoid QuPiD attack by restricting the queries with higher profile exposure. PEM measures the similarity between the user profile and the new query. If the similarity between the user profile and the query is above the threshold, the user is asked to reconsider or modify the query. As the query modification step remains outside the scope of this thesis. Therefore, future work should focus on the auto-suggested related but least similar queries mechanisms. These auto query suggestion mechanisms should also consider the user's profile and maximum privacy threshold value while suggesting the queries.

Similarly, future work could also be focused on assisting the user in setting an optimal maximum privacy threshold ($mpeT$) value. As maximum privacy threshold ($mpeT$) value is helpful in order to restrict the exposure however selection of threshold value according to the need of the user is a challenging task. Moreover, interest based threshold value of each category can also be helpful for better privacy.

Furthermore, in this thesis, we established that the PEM can effectively improve the users' privacy while they use web search engines. However, PEM uses Topics Score feature vector for both user's profile and user query. As mentioned earlier that Topics Score feature vector provides score in ten major topics against any query string. Although, Topic Score feature vector is far better as compere to the previous profiling technique i.e. text. This is one of the limitation of the feature vector (and of PEM as well) as users' interests cannot be restricted to ten topics. In future, better user profiling techniques based on ontology might be useful.

Bibliography

[1] Y. Kulizhskaya, "Snippet matching for click-tracking blocking," Ph.D. dissertation, Trinity College, 2017. [Online]. Available:. [Online]. Available: https://www.scss.tcd.ie/Doug.Leith/pubs/yana_dissertation.pdf

[2] A. Petit, "Introducing privacy in current web search engines," Ph.D. dissertation, Université de Lyon, 2017. [Online]. Available:. [Online]. Available: https://hal.inria.fr/tel-01492488v2/document

[3] Google, "Google terms of services," 2020. [Online]. Available:, (Accessed 2020-08-08). [Online]. Available: https://policies.google.com/terms?hl=en

[4] Y. Liu, K. P. Gummadi, B. Krishnamurthy, and A. Mislove, "Analyzing facebook privacy settings: user expectations vs. reality," in *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*. ACM, 2011, pp. 61–70.

[5] J. B. Earp, A. I. Antón, L. Aiman-Smith, and W. H. Stufflebeam, "Examining internet privacy policies within the context of user privacy values," *IEEE Transactions on Engineering Management*, vol. 52, no. 2, pp. 227–237, 2005.

[6] T. Rostow, "What happens when an acquaintance buys your data: A new privacy harm in the age of data brokers," *Yale Journal on Regulation*, vol. 34, p. 667, 2017.

[7] B. van Loenen, S. Kulk, and H. Ploeger, "Data protection legislation: A very hungry caterpillar: The case of mapping data in the european union," *Government Information Quarterly*, vol. 33, no. 2, pp. 338–345, 2016.

[8] E. Frantziou, "Further developments in the right to be forgotten: The european court of justice's judgment in case c-131/12, google spain, sl, google inc v agencia espanola de proteccion de datos," *Human Rights Law Review.*, vol. 14, p. 761, 2014.

[9] A. L. Newman, "What the "right to be forgotten" means for privacy in a digital age," *Science*, vol. 347, no. 6221, pp. 507–508, 2015.

[10] G. Chassang, "The impact of the eu general data protection regulation on scientific research," *ecancermedicalscience*, vol. 11, 2017.

[11] S. Gibbs, "Google to extend 'right to be forgotten' to all its domains accessed in eu," *The Guardian*, vol. 11, 2016. [Online]. Available: https://www.theguardian.com/technology/2016/feb/11/google-extend-right-to-be-forgotten-googlecom

[12] C. Nasa and S. Suman, "Evaluation of different classification techniques for web data," *International journal of computer applications*, vol. 52, no. 9, pp. 34–40, 2012.

[13] R. Khan and S. Ali, "Conceptual framework of redundant link aggregation," *Computer Science & Engineering: An International Journal*, vol. 3, no. 2, 2013.

[14] S. T. Peddinti and N. Saxena, "Web search query privacy: Evaluating query obfuscation and anonymizing networks^1," *Journal of Computer Security*, vol. 22, no. 1, pp. 155–199, 2014.

[15] T. Seymour, D. Frantsvog, and S. Kumar, "History of search engines," *International Journal of Management and Information Systems*, vol. 15, no. 4, p. 47, 2011.

[16] K. Hafner and M. Richtel, "Google resists us subpoena of search data," *New York Times*, vol. 20, p. A1, 2006. [Online]. Available: https://www.nytimes.com/2006/01/20/technology/google-resists-us-subpoena-of-search-data.html

[17] G. Pass, A. Chowdhury, and C. Torgeson, "A picture of search." in *Proceedings of the 2006 ACM InfoScale 1st International conference on Scalable information systems*, vol. 152. ACM, 2006.

[18] M. Barbaro, T. Zeller, and S. Hansell, "A face is exposed for aol searcher no. 4417749," *New York Times*, vol. 9, no. 2008, p. 8, 2006. [Online]. Available: https://www.nytimes.com/2006/08/09/technology/09aol.html

[19] E. Adar, D. S. Weld, B. N. Bershad, and S. S. Gribble, "Why we search: visualizing and predicting user behavior," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 161–170.

[20] D. J. Brenes and D. Gayo-Avello, "Stratified analysis of aol query log," *Information Sciences*, vol. 179, no. 12, pp. 1844–1858, 2009.

[21] J. Huang and E. N. Efthimiadis, "Analyzing and evaluating query reformulation strategies in web search logs," in *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, 2009, pp. 77–86.

[22] P. Heymann, G. Koutrika, and H. Garcia-Molina, "Can social bookmarking improve web search?" in *Proceedings of the 2008 International Conference on Web Search and Data Mining*. ACM, 2008, pp. 195–206.

[23] R. Jones, R. Kumar, B. Pang, and A. Tomkins, "I know what you did last summer: query logs and user privacy," in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. ACM, 2007, pp. 909–914.

[24] E. Adar, "User 4xxxxx9: Anonymizing query logs," in *Proceedings of Query Log Analysis Workshop, International Conference on World Wide Web*, 2007.

[25] B. Bi, M. Shokouhi, M. Kosinski, and T. Graepel, "Inferring the demographics of search users: Social data meets search queries," in *Proceedings of 22nd international conference on World Wide Web*. ACM, 2013, pp. 131–140.

[26] D. J. Solove, *Nothing to hide: The false tradeoff between privacy and security.* Yale University Press, 2011. [Online]. Available: https://scholarship.law.gwu.edu/cgi/viewcontent.cgi?article=2092&context=faculty_publications

[27] B. Schneier, "The eternal value of privacy," *Comment on Wired.com*, May 2006, accessed on 2018-08-08. [Online]. Available: https://www.schneier.com/essays/archives/2006/05/the_eternal_value_of.html

[28] S. B. Mokhtar, A. Boutet, P. Felber, M. Pasin, R. Pires, and V. Schiavoni, "X-search: revisiting private web search using intel sgx," in *Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference.* ACM, 2017, pp. 198–208.

[29] S. B. Mokhtar, G. Berthou, A. Diarra, V. Quéma, and A. Shoker, "Rac: A freerider-resilient, scalable, anonymous communication protocol," in *2013 IEEE 33rd International Conference on Distributed Computing Systems.* IEEE, 2013, pp. 520–529.

[30] M. G. Reed, P. F. Syverson, and D. M. Goldschlag, "Anonymous connections and onion routing," *IEEE Journal on Selected areas in Communications*, vol. 16, no. 4, pp. 482–494, 1998.

[31] D. I. Wolinsky, H. Corrigan-Gibbs, B. Ford, and A. Johnson, "Dissent in numbers: Making strong anonymity scale," in *Presented as part of the 10th USENIX Symposium on Operating Systems Design and Implementation OSDI 12)*, 2012, pp. 179–182.

[32] H. Corrigan-Gibbs and B. Ford, "Dissent: accountable anonymous group messaging," in *Proceedings of the 17th ACM conference on Computer and communications security.* ACM, 2010, pp. 340–350.

[33] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," Naval Research Lab Washington DC, Tech. Rep., 2004. [Online]. Available: https://svn-archive.torproject.org/svn/projects/design-paper/tor-design.pdf

[34] V. Toubiana, L. Subramanian, and H. Nissenbaum, "Trackmenot: Enhancing the privacy of web search," *arXiv preprint arXiv:1109.4677*, 2011. [Online]. Available: https://arxiv.org/pdf/1109.4677.pdf

[35] D. C. Howe and H. Nissenbaum, "Trackmenot: Resisting surveillance in web search," *Lessons from the Identity trail: Anonymity, privacy, and identity in a networked society*, vol. 23, pp. 417–436, 2009.

[36] J. Domingo-Ferrer, A. Solanas, and J. Castellà-Roca, "h (k)-private information retrieval from privacy-uncooperative queryable databases," *Online Information Review*, vol. 33, no. 4, pp. 720–744, 2009.

[37] J. Castellà-Roca, A. Viejo, and J. Herrera-Joancomartí, "Preserving user's privacy in web search engines," *Computer Communications*, vol. 32, no. 13-14, pp. 1541–1551, 2009.

[38] C. Romero-Tris, J. Castella-Roca, and A. Viejo, "Multi-party private web search with untrusted partners," in *International Conference on Security and Privacy in Communication Systems.* Springer, 2011, pp. 261–280.

[39] C. Romero-Tris, A. Viejo, and J. Castella-Roca, "Improving query delay in private web search," in *2011 International Conference on P2P, Parallel, Grid, Cloud and Internet Computing.* IEEE, 2011, pp. 200–206.

[40] C. Romero-Tris, A. Viejo, and J. Castellà-Roca, "Multi-party methods for privacy-preserving web search: Survey and contributions," in *Advanced Research in Data Privacy.* Springer, 2015, pp. 367–387.

[41] J. Domingo-Ferrer and M. Bras-Amorós, "Peer-to-peer private information retrieval," in *International Conference on Privacy in Statistical Databases.* Springer, 2008, pp. 315–323.

[42] J. Domingo-Ferrer, M. Bras-Amorós, Q. Wu, and J. Manjón, "User-private information retrieval based on a peer-to-peer community," *Data & Knowledge Engineering*, vol. 68, no. 11, pp. 1237–1252, 2009.

[43] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for web transactions," *ACM transactions on information and system security (TISSEC)*, vol. 1, no. 1, pp. 66–92, 1998.

[44] A. Petit, T. Cerqueus, S. B. Mokhtar, L. Brunie, and H. Kosch, "Peas: Private, efficient and accurate web search," in *2015 IEEE Trustcom/BigDataSE/ISPA*, vol. 1. IEEE, 2015, pp. 571–580.

[45] S. T. Peddinti and N. Saxena, "On the effectiveness of anonymizing networks for web search privacy," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security.* ACM, 2011, pp. 483–489.

[46] A. Petit, T. Cerqueus, A. Boutet, S. B. Mokhtar, D. Coquil, L. Brunie, and H. Kosch, "Simattack: private web search under fire," *Journal of Internet Services and Applications*, vol. 7, no. 1, pp. 2–19, 2016.

[47] Y. Lindell and E. Waisbard, "Private web search with malicious adversaries," in *International Symposium on Privacy Enhancing Technologies Symposium.* Springer, 2010, pp. 220–235.

[48] S. T. Peddinti and N. Saxena, "On the privacy of web search based on query obfuscation: a case study of trackmenot," in *International Symposium on Privacy Enhancing Technologies Symposium.* Springer, 2010, pp. 19–37.

[49] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *Proceedings of IEEE 36th Annual Foundations of Computer Science.* IEEE, 1995, pp. 41–50.

[50] B. Z. Chor, O. Goldreich, and E. Kushilevitz, "Private information retrieval," Dec. 29 1998, uS Patent 5,855,018.

[51] H. Pang, X. Ding, and X. Xiao, "Embellishing text search queries to protect user privacy," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 598–607, 2010.

[52] M. Naehrig, K. Lauter, and V. Vaikuntanathan, "Can homomorphic encryption be practical?" in *Proceedings of the 3rd ACM workshop on Cloud computing security workshop.* ACM, 2011, pp. 113–124.

[53] A. Pfitzmann and M. Waidner, "Networks without user observability," *Computers & Security*, vol. 6, no. 2, pp. 158–166, 1987.

[54] C. Romero-Tris, J. Castella-Roca, and A. Viejo, "Distributed system for private web search with untrusted partners," *Computer Networks*, vol. 67, pp. 26–42, 2014.

[55] A. Viejo and D. Sanchez, "Providing useful and private web search by means of social network profiling," in *2013 Eleventh Annual Conference on Privacy, Security and Trust.* IEEE, 2013, pp. 358–361.

[56] A. Viejo and J. Castellà-Roca, "Using social networks to distort users' profiles generated by web search engines," *Computer Networks*, vol. 54, no. 9, pp. 1343–1357, 2010.

[57] A. Viejo, J. Castella-Roca, O. Bernadó, and J. M. Mateo-Sanz, "Single-party private web search," in *2012 Tenth Annual International Conference on Privacy, Security and Trust.* IEEE, 2012, pp. 1–8.

[58] A. Viejo and D. Sánchez, "Profiling social networks to provide useful and privacy-preserving web search," *Journal of the Association for Information Science and Technology*, vol. 65, no. 12, pp. 2444–2458, 2014.

[59] M. K. Wright, M. Adler, B. N. Levine, and C. Shields, "The predecessor attack: An analysis of a threat to anonymous communications systems," *ACM Transactions on Information and System Security (TISSEC)*, vol. 7, no. 4, pp. 489–522, 2004.

[60] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE transactions on information theory*, vol. 31, no. 4, pp. 469–472, 1985.

[61] R. Cramer, G. Hanaoka, D. Hofheinz, H. Imai, E. Kiltz, R. Pass, A. Shelat, and V. Vaikuntanathan, “Bounded cca2-secure encryption,” in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2007, pp. 502–518.

[62] A. Waksman, “A permutation network,” *Journal of the ACM (JACM)*, vol. 15, no. 1, pp. 159–163, 1968.

[63] M. Jakobsson and A. Juels, “Millimix: Mixing in small batches,” DIMACS Technical report 99-33, Tech. Rep., 1999. [Online]. Available: http://www.arijuels.com/wp-content/uploads/2013/09/JJ99b.pdf

[64] Z. Cao, L. Liu, and Z. Yan, “An improved lindell-waisbard private web search scheme.” *International Journal of Network Security*, vol. 18, no. 3, pp. 538–543, 2016.

[65] A. Erola, J. Castellà-Roca, A. Viejo, and J. M. Mateo-Sanz, “Exploiting social networks to provide privacy in personalized web search,” *Journal of Systems and Software*, vol. 84, no. 10, pp. 1734–1745, 2011.

[66] D. Rebollo-Monedero, J. Forne, and J. Domingo-Ferrer, “Query profile obfuscation by means of optimal query exchange between users,” *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 5, pp. 641–654, 2012.

[67] C. E. Shannon, “A mathematical theory of communication,” *Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[68] K. Stokes and M. Bras-Amorós, “Optimal configurations for peer-to-peer user-private information retrieval,” *Computers & mathematics with applications*, vol. 59, no. 4, pp. 1568–1577, 2010.

[69] K. Stokes and M. Bras-Amoros, “On query self-submission in peer-to-peer user-private information retrieval,” in *Proceedings of the 4th International Workshop on Privacy and Anonymity in the Information Society*. ACM, 2011, p. 7.

[70] D. R. Stinson and C. Swanson, "User-private information retrieval," in *Ninth Annual Conference on Privacy, Security and Trust (PST2011)*. IEEE, 2011.

[71] C. M. Swanson and D. R. Stinson, "Extended combinatorial constructions for peer-to-peer user-private information retrieval," *arXiv preprint arXiv:1112.2762*, 2011. [Online]. Available: https://arxiv.org/pdf/1112.2762.pdf

[72] R. Khan, M. A. Islam *et al.*, "Quantification of pir protocols privacy," in *2017 International Conference on Communication, Computing and Digital Systems (C-CODE)*. IEEE, 2017, pp. 90–95.

[73] Y. Elovici, C. Glezer, and B. Shapira, "Enhancing customer privacy while searching for products and services on the world wide web," *Internet Research*, vol. 15, no. 4, pp. 378–399, 2005.

[74] Y. Elovici, B. Shapira, and A. Maschiach, "A new privacy model for hiding group interests while accessing the web," in *Proceedings of the 2002 ACM workshop on Privacy in the Electronic Society*. ACM, 2002, pp. 63–70.

[75] B. Shapira, Y. Elovici, A. Meshiach, and T. Kuflik, "Prawa privacy model for the web," *Journal of the American Society for Information Science and Technology*, vol. 56, no. 2, pp. 159–172, 2005.

[76] M. Murugesan and C. Clifton, "Providing privacy through plausibly deniable search," in *Proceedings of the 2009 SIAM International Conference on Data Mining*. SIAM, 2009, pp. 768–779.

[77] G. Chen, H. Bai, L. Shou, K. Chen, and Y. Gao, "Ups: efficient privacy protection in personalized web search," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 2011, pp. 615–624.

[78] A. Arampatzis, G. Drosatos, and P. S. Efraimidis, "Versatile query scrambling for private web search," *Information Retrieval Journal*, vol. 18, no. 4, pp. 331–358, 2015.

[79] A. Arampatzis, P. S. Efraimidis, and G. Drosatos, “A query scrambler for search privacy on the internet,” *Information RetrievalJournal*, vol. 16, no. 6, pp. 657–679, 2013.

[80] M. Juarez and V. Torra, “Dispa: An intelligent agent for private web search,” in *Advanced Research in Data Privacy.* Springer, 2015, pp. 389–405.

[81] M. Juárez and V. Torra, “Toward a privacy agent for information retrieval,” *International Journal of Intelligent Systems*, vol. 28, no. 6, pp. 606–622, 2013.

[82] M. Juarez and V. Torra, “A self-adaptive classification for the dissociating privacy agent,” in *2013 Eleventh Annual Conference on Privacy, Security and Trust.* IEEE, 2013, pp. 44–50.

[83] P. Eckersley, “How unique is your web browser?” in *International Symposium on Privacy Enhancing Technologies Symposium.* Springer, 2010, pp. 1–18.

[84] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna, “Cookieless monster: Exploring the ecosystem of web-based device fingerprinting,” in *2013 IEEE Symposium on Security and Privacy.* IEEE, 2013, pp. 541–555.

[85] C. Jackson, A. Bortz, D. Boneh, and J. C. Mitchell, “Protecting browser state from web privacy attacks,” in *Proceedings of the 15th international conference on World Wide Web.* ACM, 2006, pp. 737–744.

[86] S. Romanosky and C. Kuo, “Foxtor: Anonymous web browsing,” *Tor GUI Competition*, 2006. [Online]. Available: http://cups.cs.cmu.edu/foxtor/FoxTor-Phase2.pdf

[87] F. Saint-Jean, A. Johnson, D. Boneh, and J. Feigenbaum, “Private web search,” in *Proceedings of the 2007 ACM workshop on Privacy in electronic society.* ACM, 2007, pp. 84–90.

[88] E. Gabber, P. B. Gibbons, D. M. Kristol, Y. Matias, and A. Mayer, "Consistent, yet anonymous, web access with lpwa," *Communications of the ACM*, vol. 42, no. 2, pp. 42–47, 1999.

[89] M. Shapiro, "Structure and encapsulation in distributed systems: the proxy principle," in *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 1986, pp. 198–204.

[90] H. A. Seid and A. Lespagnol, "Virtual private network," Jun. 16 1998, uS Patent 5,768,271. [Online]. Available: https://patentimages.storage.googleapis.com/e2/7a/04/2d78306964d571/CA2202542C.pdf

[91] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84–90, 1981.

[92] O. Berthold, H. Federrath, and S. Köpsell, "Web mixes: A system for anonymous and unobservable internet access," in *Designing privacy enhancing technologies*. Springer, 2001, pp. 115–129.

[93] V. Costan and S. Devadas, "Intel sgx explained." *IACR Cryptology ePrint Archive*, vol. 2016, no. 086, pp. 1–118, 2016. [Online]. Available: http://css.csail.mit.edu/6.858/2020/readings/costan-sgx.pdf

[94] E. W. Felten and M. A. Schneider, "Timing attacks on web privacy," in *Proceedings of the 7th ACM conference on Computer and communications security*. ACM, 2000, pp. 25–32.

[95] R. Al-Rfou, W. Jannen, and N. Patwardhan, "Trackmenot-so-good-after-all," *arXiv preprint arXiv:1211.0320*, 2012. [Online]. Available: https://arxiv.org/pdf/1211.0320.pdf

[96] N. S. Evans, R. Dingledine, and C. Grothoff, "A practical congestion attack on tor using long paths." in *USENIX Security Symposium*, 2009, pp. 33–50.

[97] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on*

mathematical statistics and probability, vol. 1. Oakland, CA, USA, 1967, pp. 281–297.

[98] P. Arabie and G. De Soete, *Clustering and classification.* World Scientific, 1996.

[99] D. R. Cox, "The regression analysis of binary sequences," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 20, no. 2, pp. 215–232, 1958.

[100] R. Dietrich, M. Opper, and H. Sompolinsky, "Statistical mechanics of support vector networks," *Physical review letters*, vol. 82, no. 14, p. 2975, 1999.

[101] Y. G. Jung, M. S. Kang, and J. Heo, "Clustering performance comparison using k-means and expectation maximization algorithms," *Biotechnology & Biotechnological Equipment*, vol. 28, no. sup1, pp. S44–S48, 2014.

[102] C. Manning, P. Raghavan, and H. Schütze, "Introduction to information retrieval," *Natural Language Engineering*, vol. 16, no. 1, pp. 100–103, 2010.

[103] A. Rajaraman and J. D. Ullman, *Mining of Massive Datasets.* USA: Cambridge University Press, 2011.

[104] N. T. W. Khin and N. N. Yee, "Query classification based information retrieval system," in *2018 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, vol. 3. IEEE, 2018, pp. 151–156.

[105] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[106] Y. Freund and L. Mason, "The alternating decision tree learning algorithm," in *Proceedings of the Sixteenth International Conference on Machine Learning*, vol. 99. ACM, 1999, pp. 124–133.

[107] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "Planetlab: an overlay testbed for broad-coverage services," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 3, pp. 3–12, 2003.

[108] J. Novak, P. Raghavan, and A. Tomkins, "Anti-aliasing on the web," in *Proceedings of the 13th international conference on World Wide Web.* ACM, 2004, pp. 30–39.

[109] E. Balsa, C. Troncoso, and C. Diaz, "Ob-pws: Obfuscation-based private web search," in *2012 IEEE Symposium on Security and Privacy.* IEEE, 2012, pp. 491–505.

[110] S. Ye, F. Wu, R. Pandey, and H. Chen, "Noise injection for search privacy protection," in *2009 International Conference on Computational Science and Engineering*, vol. 3. IEEE, 2009, pp. 1–8.

[111] A. Gervais, R. Shokri, A. Singla, S. Capkun, and V. Lenders, "Quantifying web-search privacy," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security.* ACM, 2014, pp. 966–977.

[112] P. Jaccard, "The distribution of the flora in the alpine zone. 1," *New phytologist*, vol. 11, no. 2, pp. 37–50, 1912.

[113] A. Singhal *et al.*, "Modern information retrieval: A brief overview," *IEEE Data Engineering Bulletin*, vol. 24, no. 4, pp. 35–43, 2001.

[114] L. R. Dice, "Measures of the amount of ecologic association between species," *Ecology*, vol. 26, no. 3, pp. 297–302, 1945.

[115] C. E. Shannon, "A mathematical theory of communication," *Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[116] C. Diaz, S. Seys, J. Claessens, and B. Preneel, "Towards measuring anonymity," in *International Workshop on Privacy Enhancing Technologies.* Springer, 2002, pp. 54–68.

[117] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[118] D. Rebollo-Monedero and J. Forné, "Optimized query forgery for private information retrieval," *IEEE Transactions on Information Theory*, vol. 56, no. 9, pp. 4631–4642, 2010.

[119] J. Parra-Arnau, D. Rebollo-Monedero, and J. Forné, "Measuring the privacy of user profiles in personalized information systems," *Future Generation Computer Systems*, vol. 33, pp. 53–63, 2014.

[120] S. V. Stehman, "Selecting and interpreting measures of thematic classification accuracy," *Remote sensing of Environment*, vol. 62, no. 1, pp. 77–89, 1997.

[121] Y. Yang, "An evaluation of statistical approaches to text categorization," *Information retrieval*, vol. 1, no. 1-2, pp. 69–90, 1999.

[122] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuéllar, P. H. Drielsma, P.-C. Héam, O. Kouchnarenko, J. Mantovani *et al.*, "The avispa tool for the automated validation of internet security protocols and applications," in *International conference on computer aided verification.* Springer, 2005, pp. 281–285.

[123] B. Blanchet, "An efficient cryptographic protocol verifier based on prolog rules," in *Proceedings of the 14th IEEE Workshop on Computer Security Foundations*, ser. CSFW 01. USA: IEEE Computer Society, 2001, p. 82.

[124] R. Khan, M. Ullah, and M. A. Islam, "Revealing pir protocols protected users," in *2016 Sixth International Conference on Innovative Computing Technology (INTECH).* IEEE, 2016, pp. 535–541.

[125] M. Alemzadeh and F. Karray, "An efficient method for tagging a query with category labels using wikipedia towards enhancing search engine results," in *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology-Volume 01.* IEEE Computer Society, 2010, pp. 192–195.

[126] J. R. Quinlan, *C4. 5: programs for machine learning.* Morgan Kaufmann Publishers Inc. San Mateo , California, 2014.

[127] M. Sumner, E. Frank, and M. Hall, "Speeding up logistic model tree induction," in *European conference on principles of data mining and knowledge discovery.* Springer, 2005, pp. 675–683.

[128] R. Kohavi, "The power of decision tables," in *European conference on machine learning.* Springer, 1995, pp. 174–189.

[129] W. W. Cohen, "Fast effective rule induction," in *Machine Learning Proceedings 1995.* Elsevier, 1995, pp. 115–123.

[130] R. C. Holte, "Very simple classification rules perform well on most commonly used datasets," *Machine learning*, vol. 11, no. 1, pp. 63–90, 1993.

[131] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," *Machine learning*, vol. 6, no. 1, pp. 37–66, 1991.

[132] J. G. Cleary and L. E. Trigg, "K*: An instance-based learner using an entropic distance measure," in *Machine Learning Proceedings 1995.* Elsevier, 1995, pp. 108–114.

[133] T. H. Lee and Y. Yang, "Bagging binary and quantile predictors for time series," *Journal of econometrics*, vol. 135, no. 1-2, pp. 465–497, 2006.

[134] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining.* ACM, 2016, pp. 785–794.

[135] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Machine learning*, vol. 29, no. 2-3, pp. 131–163, 1997.

[136] N. Midha and V. Singh, "Classification of e-commerce products using reptree and k-means hybrid approach," in *Big Data Analytics.* Springer, 2018, pp. 265–273.

[137] G. H. John and P. Langley, "Estimating continuous distributions in bayesian classifiers," in *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence.* Morgan Kaufmann Publishers Inc., 1995, pp. 338–345.

[138] I. Rish *et al.*, “An empirical study of the naive bayes classifier,” in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, 2001, pp. 41–46.

[139] Y. Y. Chen, Y. H. Lin, C. C. Kung, M.-H. Chung, and I. H. Yen, “Design and implementation of cloud analytics-assisted smart power meters considering advanced artificial intelligence as edge analytics in demand-side management for smart homes,” *Sensors*, vol. 19, no. 9, p. 2047, 2019.

[140] A. Graves, “Long short-term memory,” in *Supervised sequence labelling with recurrent neural networks.* Springer, 2012, pp. 37–45.

[141] G. C. Cawley and N. L. Talbot, “On over-fitting in model selection and subsequent selection bias in performance evaluation,” *The Journal of Machine Learning Research*, vol. 11, pp. 2079–2107, 2010.

[142] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning.* Springer, 2013, vol. 112.

[143] Z. Reitermanova, “Data splitting,” in *Proceedings of the 19th Annual Conference of Doctoral Students - WDS’10*, vol. 10, 2010, pp. 31–36.

[144] B. S. Everitt, S. Landau, M. Leese, and D. Stahl, “Miscellaneous clustering methods,” *Cluster analysis*, pp. 215–255, 2011.

[145] F. Bravo-Marquez, E. Frank, B. Pfahringer, and S. M. Mohammad, “Affectivetweets: a weka package for analyzing affect in tweets,” *Journal of Machine Learning Research*, vol. 20, pp. 1–6, 2019.

[146] B. Wang, A. Wang, F. Chen, Y. Wang, and C.-C. J. Kuo, “Evaluating word embedding models: Methods and experimental results,” *APSIPA transactions on signal and information processing*, vol. 8, 2019.

[147] S. Lang, F. Bravo-Marquez, C. Beckham, M. Hall, and E. Frank, “Wekadeeplearning4j: A deep learning package for weka based on deeplearning4j,” *Knowledge-Based Systems*, vol. 178, pp. 48–50, 2019.

[148] L. Hamers, Y. Hemeryck, G. Herweyers, M. Janssen, H. Keters, R. Rousseau, and A. Vanhoutte, “Similarity measures in scientometric research: The jaccard index versus salton’s cosine formula.” *Information Processing and Management*, vol. 25, no. 3, pp. 315–18, 1989.

[149] S. R. Ghorpade and B. V. Limaye, *A course in multivariable calculus and analysis*. Springer Science & Business Media, 2010.

[150] D. Olszewski, “Fraud detection in telecommunications using kullback-leibler divergence and latent dirichlet allocation,” in *International Conference on Adaptive and Natural Computing Algorithms*. Springer, 2011, pp. 71–80.

[151] Y. Xu, K. Wang, B. Zhang, and Z. Chen, “Privacy-enhancing personalized web search,” in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 591–600.

[152] F. Qiu and J. Cho, “Automatic identification of user interest for personalized search,” in *Proceedings of the 15th international conference on World Wide Web*, 2006, pp. 727–736.

[153] N. Landwehr, M. Hall, and E. Frank, “Logistic model trees,” *Machine learning*, vol. 59, no. 1-2, pp. 161–205, 2005.

[154] B. Kamiński, M. Jakubczyk, and P. Szufel, “A framework for sensitivity analysis of decision trees,” *Central European journal of operations research*, vol. 26, no. 1, pp. 135–159, 2018.

Appendix A

Stopwords

about	ask	by	each	four
above	asked	c	early	from
across	asking	came	either	full
after	asks	can	end	fully
again	at	cannot	ended	further
against	away	case	ending	furthered
all	b	cases	ends	furthering
almost	back	certain	enough	furthers
alone	backed	certainly	even	g
along	backing	clear	evenly	gave
already	backs	clearly	ever	general
also	be	come	every	generally
although	became	could	everybody	get
always	because	d	everyone	gets
among	become	did	everything	give
an	becomes	differ	everywhere	given
and	been	different	f	gives
another	before	differently	face	go
any	began	do	faces	going
anybody	behind	does	fact	good
anyone	being	done	facts	goods
anything	beings	down	far	got
anywhere	best	down	felt	great
are	better	downed	few	greater
area	between	downing	find	greatest
areas	big	downs	finds	group
around	both	during	first	grouped
as	but	e	for	grouping

groups	later	noone	presented	someone
h	latest	not	presenting	something
had	least	nothing	presents	somewhere
has	less	now	problem	state
have	let	nowhere	problems	states
having	lets	number	put	still
he	like	numbers	puts	still
her	likely	o	q	such
here	long	of	quite	sure
herself	longer	off	r	t
high	longest	often	rather	take
high	m	old	really	taken
high	made	older	right	than
higher	make	oldest	right	that
highest	making	on	room	the
him	man	once	rooms	their
himself	many	one	s	them
his	may	only	said	then
how	me	open	same	there
however	member	opened	saw	therefore
i	members	opening	say	these
if	men	opens	says	they
important	might	or	second	thing
in	more	order	seconds	things
interest	most	ordered	see	think
interested	mostly	ordering	seem	thinks
interesting	mr	orders	seemed	this
interests	mrs	other	seeming	those
into	much	others	seems	though
is	must	our	sees	thought
it	my	out	several	thoughts
its	myself	over	shall	three
itself	n	p	she	through
j	necessary	part	should	thus
just	need	parted	show	to
k	needed	parting	showed	today
keep	needing	parts	showing	together
keeps	needs	per	shows	too
kind	never	perhaps	side	took
knew	new	place	sides	toward
know	new	places	since	turn
known	newer	point	small	turned
knows	newest	pointed	smaller	turning
l	next	pointing	smallest	turns
large	no	points	so	two
largely	nobody	possible	some	u

Appendix B

Complementary Experiments on QuPiD Attack

In this appendix, we present further experiments we performed with QuPiD attack. These experiments are performed to evaluate the effectiveness of QuPiD Attack with different classification algorithms under variable user profile size. The selection of algorithms is made from different families of classification algorithms (such as Tree-based, Rule-based, Ensemble Learning, Lazy-Learner etc). We first present the experimentation setup in detail with description of all selected algorithms. The performance of QuPiD attack under variable user profile size and different algorithms is evaluated in terms of *precision*, *recall*, *f-measure*, and true positive rate and discussed in last section of this appendix.

B.1 Experimentation Preliminaries

For experimentation we used AOL-100 data set which is composed of 100 randomly selected active users from the released AOL dataset. The user selection criteria and dataset description is available in section 3.4.4. The dataset is composed of 13 attributes: user ID, query, date and time of the query along with ten acquired attributes that are used to classify the user query into 10 major topics. The

Table B.1: AOL-100 Dataset Properties.

Total queries	36,389,567
Total users	657,426
Unique queries	10,154,742
Attributes	5 (AnonID, Query, Query Time, Item Rank, Click URL)
Time duration	01 March, 2006 - 31 May, 2006

10 major topics are Health, Recreations, Arts, Home, Business, Society, Games, Sciences, Computers, and Science.

These 100 selected users are further divided into 5 groups (20 users in each group) based on the size of the user's profile in order to evaluate the performance of the selected algorithms with different profile size. The summary of the selected users' dataset is given in the Table B.1.

B.2 Machine Learning Algorithms Employed

The machine learning algorithms are a very integral part of the proposed model as they are used for the building of the classification model. In previous studies, Peddinti et al. [14, 45] and Petit et al. [2] used Random forest, AD Tree, Zero R, Regression, and SVM algorithms for the classification of the data queries. In both studies the classification model was bi-class, i.e., the query is machine or user-generated. Moreover, the model was built based on two attributes like query and assigned label. In our work, however, the classification model is multiclass i.e., in the testing data the model will decide which query belongs to which user and model is based on twelve attributes. Initially we select 10 off-the-shelf (default setting) classification algorithms form different families such as J48 [123] and Logistic Model Tree (LMT) [124] from the tree-based family, Decision Table [125], JRip [126] and OneR [127] from Rule-based family, IBK [128] and KStar [129] from Lazy-Learner family, Bagging [130] and XGBoost [134] from Meta-heuristic (Ensemble Learning) family and Naïve Bayes [132] from Bayes family. Rep Tree [133] and Regression are used as base classifiers for Begging algorithm. However due to poor performance many algorithms, we select the top five algorithms i.e. Naïve

Bayes, IBk, Bagging, XGBoost, and J48. In addition to Bagging, XGBoost is used from the ensemble learning algorithm category. XGBoost is machine learning algorithm that uses gradient boosting technique in order to improve the performance of weak decision tree based prediction models [134]. The performance comparison of all 10 selected algorithms is available in in this section. Parameters of all selected machine learning algorithms are shown in Table B.2. A brief introduction of each classifier is provided below:

- **J.48:** J.48 is tree-based classifier which is based on the C4.5 algorithm [123]. It's a statistical classifier which creates a decision tree-based on information entropy. It is considered as the most prominent and most widely used algorithm for machine learning activities.
- **Logistic Model Tree (LMT):** Logistic Model Tree is a supervised learning classification algorithm that combines decision tree and logistic regression [153]. This algorithm use standard decision tree structure build by C4.5 algorithm with logistic regression functions at the nodes. The tree is then prune with well-known CART (Classification and Regression for Trees) algorithm [124].
- **Decision Table:** In Decision Table the decision information is expressed in the form of Decision Tree [154]. The decision tree is created based on the events recorded in the past (training data) using if-then-else strategy.
- **JRip:** JRip is a rule based supervised learning algorithm use to create fast and accurate classification model [126]. JRip is the enhanced and more efficient version of IREP (Incremental Reduced Error Pruning) classification algorithm.
- **One R:** One Rule (One R) is a rule-based algorithm which creates one rule for each predictor. After the generation of all the rules, the rule with the smallest total errors is then nominated as ”one rule” [127].
- **IBK:** This algorithm belongs to the distance-based group of the classifiers. It uses k-nearest neighbor classifier to find the distance between two vectors

[128]. In this research, the algorithm used to a searching nearest neighbor is "Linear-NN-Search" which works on "Euclidean Distance". The rest of the parameters remain default.

- **KStar:** KStar [129] is an instance-based supervised learning algorithm that belongs to the Lazy-Learner family. Instance-based learners classify an instance by comparing it to a database of pre-classified examples. KStar used entropy-based distance function for classification.

- **Bagging:** Bootstrap Aggregation (Bagging) is meta-heuristic algorithms that use different machine learning algorithms to achieve better prediction [130]. It divides the training data into small datasets and creates classifiers for each dataset based on selected classifier. The results of all small datasets are then combined using average or majority voting or other methods. For this research, REPTree algorithm is used as base learner.

- **XGBoost:** XGBoost stands for extreme gradient boost is an ensemble machine learning algorithm, used to deal with classification and regression problems. XGBoost is evolved from decision trees algorithm that uses gradient boosting techniques to minimize the error in sequential prediction tree [134]. Extreme Gradient Boost is the evolved version of Gradient boosting technique that offer parallel processing, tree-pruning, and missing values handling techniques.

- **Bayes Net:** This algorithm belongs to the probabilistic class of the classifiers which predicts the chance of occurring an event-based analysis of past events [134]. The popular applications of this classifiers are medical diagnostics, spam identification and text classification [135]. In this research, the most generic version is used with default parameters.

Where **Batch Size** shows the number of instances to process in case of batch prediction. **Kernel Estimator** is a weighting function used in non-parametric technique. **KNN** value shows the number of neighbors to used for calculation. **NN Search Algorithm** is the Algorithm used to find the nearest neighbor. **Window**

TABLE B.2: Parameters of all Selected Classifiers used in Appendix B for QuPiD Attack.

Algorithm	**Parameters**
J48	Confidence Factor = 0.25 Seed =1 No of Folds = 3 Batch size = 100 Pruning = False
LMT	Batch Size = 100 Fast Regression = True Minimum No of Instances = 15 Weight Trim Beta = 0.0
Decision Table	Batch Size =100 Cross Validation = 1 Attribute Selection = Best First

J Rip	Batch Size 100
	Pruning Folds = 3
	Min Instance weight= 2.0
	Optimization = 2
	Seed =1
	Pruning = True

One R	Batch Size =100 Min Bucket Size = 6
IBk	Batch Size =100 KNN value =1 NN search algorithm = Linear NN Search (Euclidean Distance) Window Size =0
K Star	Batch Size =100 Global Blend = 20 Missing Mode = Ave Column Entropy
Bagging	Bag Size= 100 Batch Size =100 Iterations = 10 Classifier = REP Tree Seed = 1 Num Folds = 3 Min Variance = 0.001 Max Depth = -1

XG Boost	Verbosity = 1
	Iteration = 250
	Learning Rate = 0.3
	Seed = 1
	Sub Sample = 0.5

Bayes Net	Batch size = 100
	Debug = False
	Estimator = Simple Estimator
	Search Algorithm = Hill Climbing Algorithm

Size is used to define maximum number of training instances maintained. **Bag Size** is used to define the size of bag for training data. **Iterations** parameter is used to specify the number of iterations to be performed by the algorithm. **Classifier** parameter specifies the base Classifier used by in bagging classifier. **Seed** parameter is used for randomize the data. **Num folds** specifies the amount of data used for pruning. **Max Depth** shows the maximum tree depth (-1 shows no restriction). **Verbosity** shows whether to print the early stopping information or not. **Learning Rate** shows that how quickly the error is corrected from each tree to the next. **Sub Sample** means that XGBoost (Extreme Gradient Boost) would randomly sample half of the training data prior to growing trees to avoid the problem of over-fitting. **Confidence Factor** is used for tree pruning. **No of folds** specifies the amount of data used to reduced-error pruning. **Type** shows the class of Artificial Neural Network used. **Layer** specifies the type of layer used. **Activation Function** of a node defines the output of that node given an input or set of inputs. **Gate Activation Function** defines the activation function to use for the gates. **No of Epochs** is the number of complete passes through the training dataset.

B.3 Results and Discussion

These experiments are performed to evaluate the effectiveness of QuPiD Attack with different classification algorithms under variable user profile size. The selection of algorithms is made from different families classification algorithms. The experiments are performed on 100 users dataset distributed in five groups in order to observe the impact of the size of training on the accuracy of results. For each UoI, we measured *precision*, *recall*, and *true positive* percentage of correctly classified queries from an anonymized log.

Table B.3 shows the true positive percentage of the queries of UoI. According to Table B.3, all algorithms correctly identified more than 89% queries of 2 users except OneR. One R correctly identified 80% to 90% queries of 4 users. Overall

TABLE B.3: Percentage of Users in a Group based on *True Positive* Values.

True Positive Percentage Bands	Tree-Based		Rule-Based			Lazy Learner		Ensemble		Bayesian
	J48	LMT	DT	JRip	OneR	IBk	KStar	Bagging	XGBoost	Bayes Net
100%-90%	2	2	2	2	0	2	2	2	2	2
90%-80%	2	2	2	2	4	2	0	2	2	2
80%-70%	4	2	4	2	0	4	4	4	4	4
70%-60%	4	8	4	2	6	4	6	4	4	2
60%-50%	14	2	4	6	0	24	16	18	18	14
50%-40%	26	28	24	4	10	18	22	20	20	20
Below 40%	48	56	60	82	80	46	50	50	50	56

IBK correctly identified more than 50% queries of 36 users followed by Bagging and KStar with 30 and 28 users respectively.

As mentioned earlier, the experiments are performed on 100 users dataset distributed in five groups in order to observe the impact of the size of training on the accuracy of results. Table B.4 shows the comparison of the performance of all algorithms with a variation of the training dataset size. The performance of each algorithm is measured in *precision*, *recall* and average *f-measure*. The results shows that IBk and KStar associated more than 40% queries to the correct user with the *precision* of above 60% in all cases. While Bagging, J48, Decision Table (DT), and Bayes Net associate more than 25% queries to the correct user with the *precision* of above 60% in all cases. From the perspective of the size of training dataset, it is slightly difficult to draw a conclusion about its effect on accuracy due to ProQSim Effect (section 3.6). Almost every algorithm shows irregular behavior with a variation in the training dataset size. For the first three groups, the performance of IBK, J48, KStar, and LMT (Logistic Model Tree) is observed more accurate. However, unexpectedly, the rate of *recall* drops for the last two groups. The results of *precision* and *recall* are plotted in Fig. B.1 and B.2 respectively.

Table B.4: Group wise *precision* and *recall* with Different Algorithms.

Group			Group 1	Group 2	Group 3	Group 4	Group 5
Tree-Based	J48	Precision	0.68	0.71	0.75	0.72	0.72
		Recall	0.37	0.40	0.44	0.36	0.43
	LMT	Precision	0.69	0.70	0.70	0.75	0.72
		Recall	0.36	0.38	0.43	0.33	0.42
Rule-Based	Decision Table	Precision	0.86	0.89	0.90	0.79	0.79
		Recall	0.33	0.32	0.41	0.34	0.41
	JRip	Precision	0.85	0.80	0.85	0.77	0.78
		Recall	0.25	0.23	0.32	0.23	0.34
	OneR	Precision	0.46	0.39	0.48	0.46	0.51
		Recall	0.21	0.17	0.27	0.25	0.35
Lazy Learner	IBK	Precision	0.74	0.78	0.83	0.78	0.77
		Recall	0.42	0.44	0.48	0.38	0.45
	KStar	Precision	0.75	0.78	0.77	0.76	0.72
		Recall	0.36	0.40	0.44	0.35	0.72
Ensemble	Bagging	Precision	0.77	0.74	0.78	0.79	0.73
		Recall	0.37	0.41	0.45	0.36	0.44
	XGBoost	Precision	0.76	0.75	0.78	0.80	0.70
		Recall	0.37	0.40	0.46	0.37	0.44
Bayesian	Bayes Net	Precision	0.77	0.71	0.77	0.78	0.69
		Recall	0.32	0.36	0.42	0.33	0.44

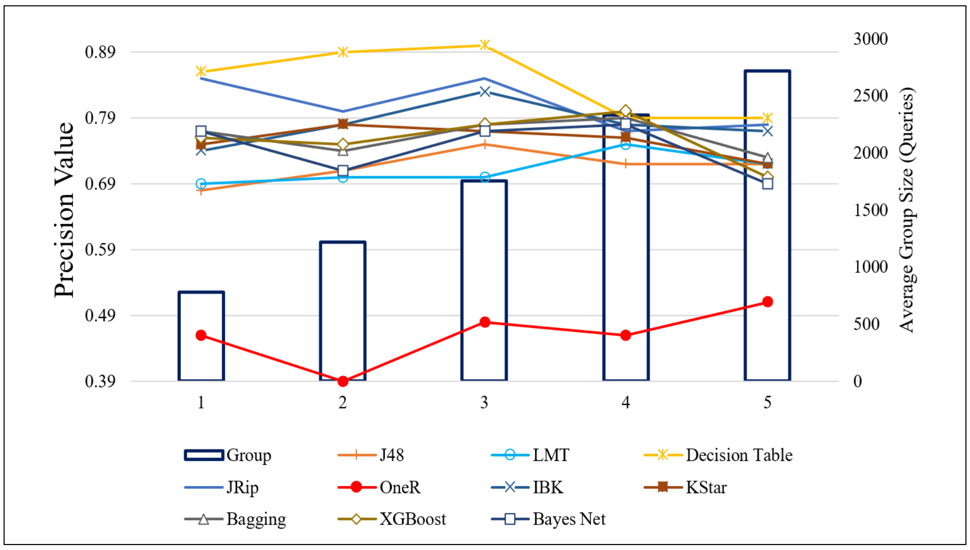


Figure B.1: Group wise *precision* with Selected Classification Algorithms.

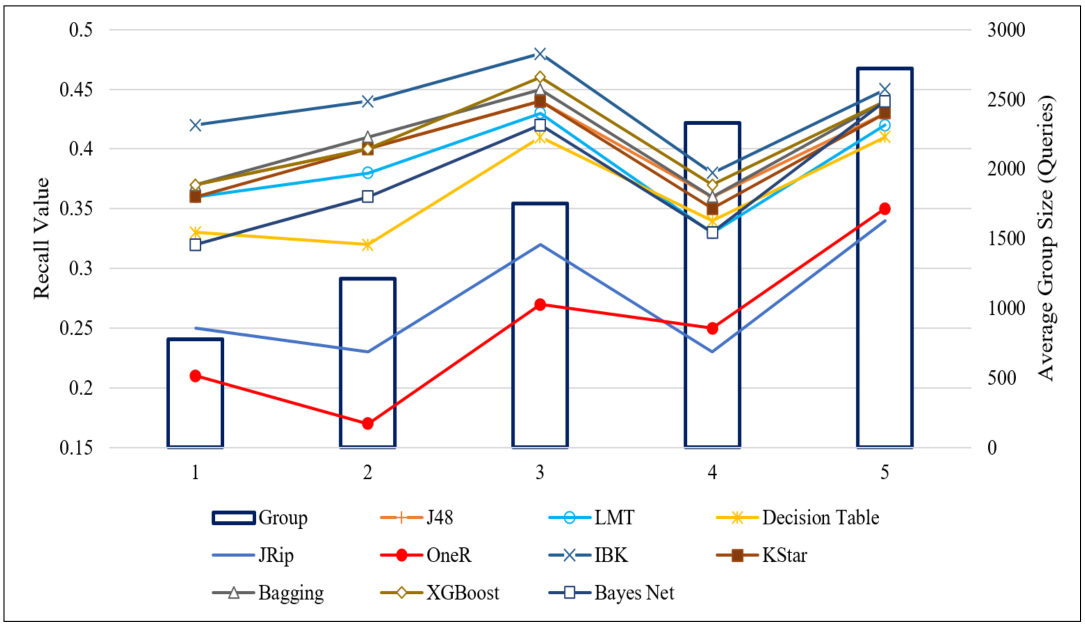


FIGURE B.2: Group wise *recall* with Selected Classification Algorithms.

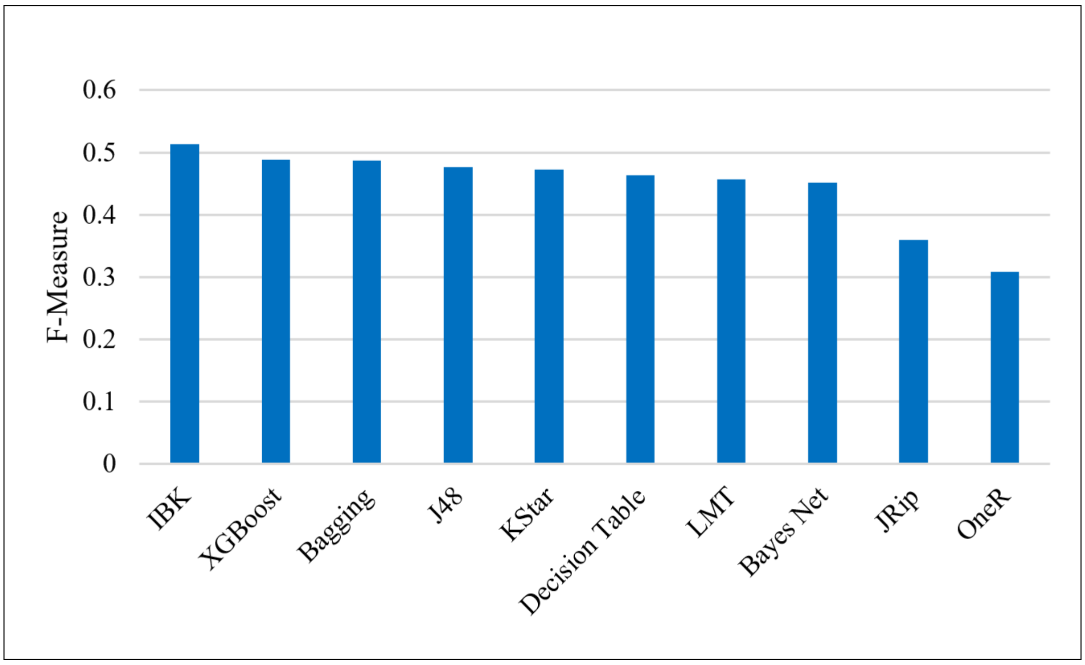


FIGURE B.3: Average *f-measure* of all Selected Classification Algorithms.

Overall, IBK and Bagging associated 45.1% and 43% queries to the correct user with above 70% *precision*. While J48, KStar, and LMT associated 42.2%, 41.7% and 40.6% queries to the correct user with the *precision* of 70.9%, 73.5%, and 70.2%. The top three algorithms in terms of *f-measure* (trade-off between *precision* and *recall*) are IBK, Bagging, and J48 with the score of 0.514, 0.487 and 0.477 respectively. The results of average *f-measure* are plotted in Fig. B.3.