Data Security Techniques and Comparison of Differential Privacy Techniques in Bioinformatics

Received: 29 January 2023; Accepted: 16 February 2023 *Research Article*

Nilgün İncereis
Department of Computer Engineering, Distance Education Application and Research Center
Istanbul Okan University, Bartın University
Istanbul, Bartın, Türkiye
niincereis@stu.okan.edu.tr
nincereis@bartin.edu.tr
ORCID: 0000-0001-5508-8159

Hilal Çakır
Deparment of Computer Engineering
Istanbul Okan University
Istanbul, Türkiye
hilcakir@stu.okan.edu.tr
ORCID: 0000-0002-5378-7930

Bekir Tevfik Akgün
Software Development
Yeditepe University
Istanbul, Türkiye
bekirtevfik.akgun@yeditepe.edu.tr
ORCID: 0000-0002-9726-1340

***Abstract*—Bioinformatics data is data containing information about biological systems and processes. This data can include genomic data, proteomic data, metabolic data, and similar data. The processing and analysis of bioinformatics data aims to achieve important goals such as conducting scientific research and improving healthcare systems. Data security of bioinformatics data ensures the security of data during processing and analysis as well as protecting individual privacy. In this study, five of the known techniques for data security in bioinformatics have been studied. These techniques include: data anonymization, data masking, data encryption, and role-based access control, and differential privacy. In this study, it is aimed to create functions for the above-mentioned data security techniques by using the dataset obtained from 1000 patients with lung cancer, and to anonymize the dataset by using Laplacian, Gaussian and Exponential mechanisms from differential privacy techniques. Looking at various comparison parameters from the differential privacy techniques, it is concluded that the Laplacian technique strikes the best balance between privacy and utility as it provides the highest privacy guarantee and accuracy, as well as the lowest noise and robustness.**

Keywords*— *Bioinformatics, data security, data anonymization, data masking, data encryption, role-based access control, differential privacy.

I. INTRODUCTION

Today, bioinformatics is a field that integrates computer science and biology. Computer technology and software [1], [2], [3] are used in this field to collect, store, analyze and understand biological data. By analyzing genetic data, bioinformatics research helps to understand the composition and capabilities of human and animal genomes. This topic can also be used for modeling and simulation [4], [5] of many biological systems.

In other words, bioinformatics is an active research area that aims to develop intelligent systems for molecular biology analysis. Many methods based on formal linguistic theory, statistical theory, and learning theory have been developed to model and analyze biological sequences such as DNA, RNA, and proteins. In particular, grammatical inference methods are expected to find some grammatical structures hidden in biological sequences. A study [1] provides an overview of a number of grammatical approaches to biological sequence analysis and related research, focusing on learning stochastic grammars from biological sequences and predicting their functions based on learned stochastic grammars.

The science of bioinformatics focuses on using information technology to process, store, and analyze biological data. The methods utilized to store and safeguard this data are referred to as data security [6]. These methods are crucial since bioinformatics data frequently contains private and confidential information. Bioinformatics data, for instance, may comprise details such as genetic, clinical, or biological samples. In addition to ensuring that this data is utilized precisely and consistently, data security is crucial for safeguarding the privacy and security of personal information.

Because it combines computer science, molecular biology, and genetics, bioinformatics encompasses issues like genomics, proteomics, and metabolomics and enables the analysis of this data using techniques like artificial intelligence, machine learning, and data mining. However, the security of this data is crucial since it is frequently vital data. The protection of data from unlawful access, alteration, or destruction is ensured by data security. Methods including data anonymization [7], [8], data masking [9], data encryption [10] and role-based access control [11], and differential privacy [12], [13] are examples of data security strategies used in bioinformatics. These methods protect the accuracy and integrity of your data while ensuring its security.

In this study, an introduction to the field of bioinformatics was made in Chapter I. In Chapter II, the importance of bioinformatics data is explained. Chapter III describes 5 of the known data security techniques for bioinformatics. Chapter IV outlines the advantages of known data security techniques in bioinformatics. In Chapter V, applications of known data security techniques are explained.

II. THE IMPORTANCE OF BIOINFORMATICS DATA

In this section, bioinformatics data and its privacy, a brief history of bioinformatics, and present and future of bioinformatics will be discussed.

A. Bioinformatics Data and Its Privacy

Bioinformatics data is data that contains information about biological systems. This data may include data such as genetic information in an individual's DNA (genomic data) [14], structure of proteins (proteomic data) [15], and biochemical reactions (metabolic data). It targets important purposes such as processing and examining bioinformatics

data, conducting scientific research and improving health systems.

During the processing and examination of bioinformatics data, it is important to protect the privacy of individuals and ensure the security of the data. To promote scientific advancement and cut costs, there have been greater efforts to share research data. It is now obvious that no scientist can promise complete anonymity, and it is also becoming more widely accepted that research will be more successful if scientists have more knowledge about the subjects, and that being recognizable has certain advantages [16]. However, Concern over the privacy of medical information has long been expressed by patients and study participants. In a study [17], 92% of the participants preferred to be requested permission before having their health information used for anything other than medical care, and 83% of them wanted to know the specifics of the research before consenting to have their health records used. According to the study, there are several topics that are particularly delicate, such as lists of previous procedures and current drugs, as well as family medical history, genetic abnormalities, mental illness, and drug or alcohol-related occurrences. There are ethical concerns regarding the ability of subjects with cognitive impairment to provide informed consent or people with addiction to take part in research that administer medicines of dependence [16].

B. History of Bioinformatics

More than 50 years ago, when desktop computers were merely a theory and DNA sequencing was not yet possible, bioinformatics was just beginning. Early in the 1960s, the use of computer techniques to protein sequence analysis lay the groundwork for bioinformatics (notably, de novo sequence assembly, biological sequence databases and substitution models). DNA analysis later developed as a result of developments in both computer science and molecular biology, which resulted in increasingly powerful and compact computers as well as new software that was better suited to handle bioinformatics tasks. These developments allowed for easier manipulation and sequencing of DNA. Significant advancements in sequencing technology and cost-cutting measures during the 1990s and 2000s led to an exponential growth of data [18].

C. Present and Future of Bioinformatics

Only a few of the current applications for bioinformatics include gene expression analysis [19], genome annotation [20], protein structure prediction [15], and drug development. It is also being used to identify genetic risk factors for diseases and to develop personalized medical methods [21].

It is anticipated that bioinformatics will continue to be essential to biology and medicine in the future. The analysis and interpretation of biological data using machine learning and artificial intelligence methods is one area of great interest. These methods could significantly increase our comprehension of biological systems and quicken the pace of scientific discovery [22].

The development of tools and methods for analyzing and interpreting data from large-scale, multi-omic studies [23], which involve the simultaneous analysis of multiple types of biological data, such as genomics [14], transcriptomics [24], and proteomics [15], is another area of focus for future bioinformatics research.

III. KNOWN DATA SECURITY TECHNIQUES FOR BIOINFORMATICS

There are many data security techniques for bioinformatics. In this study, data anonymization, data encryption, masking and role-based access control methods are given below.

A. Data Anonymization

Data anonymization [7] is the process of removing personal identifying information from data sets so that the individuals who are the subject of the data cannot be identified. This is often done to protect the privacy of individuals and to comply with laws and regulations that require personal data to be kept confidential. In the field of bioinformatics, data de-identification is important because it enables researchers to share data and collaborate without compromising the privacy of individuals [16].

By incorporating or integrating the data of study participants into bigger repositories, the practice of building massive databases has grown more widespread. As a result, sufficient sample numbers may be used for analysis of the kind seen in genome-wide association studies. The initial permission procedure for the smaller research frequently did not include the concept of data sharing, which is the joining of smaller datasets into bigger, independent datasets. Personal identifiers must be deleted from the data to ensure participant anonymity, and a coded link may be maintained between a participant's data and identity to enable potential therapeutic updates, longitudinal epidemiology studies, or the return of specific study findings [25].

There are several ways to anonymize bioinformatics data, including the following:

- *Remove personal identifiers:* Personal identifiers such as names, addresses, and social security numbers should be removed from the dataset.
- *Use anonymized identifiers:* Instead of using personal identifiers, anonymized identifiers can be used to identify records. These identifiers should not be linked to any personal information.
- *Remove or modify sensitive data:* Sensitive data such as medical diagnoses or genetic information should be removed or modified to prevent re-identification of individuals.
- *Use data aggregation and generalization:* Aggregating data and using general categories (e.g., age ranges) rather than specific values can further reduce the risk of re-identification.
- *Apply statistical techniques:* Statistical techniques such as noise injection and k-anonymization can be used to further de-identify the data.

To effectively protect the privacy of individuals, it's crucial to carefully consider the level of anonymization required for a given dataset and to use a combination of these techniques.

There are some libraries which can be used for data anonymization:

- *pynonymizer*: It is a Python library [26] which is used worldwide to convert private production database dumps into anonymous copies.
- *Faker*: It is a Python package [27] that can create fictitious data for the biological private data.

B. Data Masking

Data masking [9] is concealing or changing certain data in a data set. This procedure is frequently carried out to safeguard the data's confidentiality or to check its correctness. There are several ways to alter and recreate data sets using data masking techniques. For instance, it is possible to remove specific data from a data collection or disguise the names of the individuals who are included in it. In this manner, both the data set and its accuracy and privacy are preserved.

In the realm of bioinformatics, there are several libraries and tools that may be used for data masking. These may consist of:

- *py-anon*: This Python module [28] offers a straightforward and adaptable framework for anonymizing and masking sensitive data.
- *DataShield*: This R package [29] offers a selection of safe calculations and sensitive data masking functions, and it is built to operate in a distributed setting.
- *Anonymizer*: A selection of tools for data masking and anonymization, as well as supports for data modification, generalization, and suppression, are provided by this Java package [30].
- *Data Masker*: This [31] is a command line tool that provides data masking functionality.
- *DataRobot*: This [32] is a software platform for data science and also provides data masking functionality.

Using these frameworks and tools, data masking strategies may be applied to massive datasets in a variety of forms, including CSV, JSON, and SQL databases. It may also make use of various encryption keys and techniques. These libraries include pre-built masking routines and masking rules, as well as the ability to employ masking techniques programmatically.

It is crucial to consider that each library or tool has certain benefits, functions, and capabilities. The particular use case and requirements also determine the library or tools it should be used.

C. Data Encryption

Particularly in the field of bioinformatics, where sensitive and private information is frequently kept and transmitted, data encryption [10] is a significant component of data security. Data is protected from potential breaches and misuse via encryption, which ensures that it cannot be viewed or understood by unauthorized parties.

Various techniques, such as hash functions, symmetric key encryption, and asymmetric key encryption, can be used to encrypt bioinformatics data. Data is encrypted and decrypted using the same shared key in symmetric key encryption. This method is quick and efficient, but it necessitates a secure key exchange between the parties, which could be challenging. Asymmetric key encryption, also known as public key encryption, encrypts and decrypts data using a set of two keys: a public key and a private key. The public key is used to encrypt the data, and the private key is used to decrypt it. This method is more secure than symmetric key encryption, despite being slower and requiring more computing power. In contrast, hash functions do not encrypt data using keys. Instead, a mathematical procedure is used to transform the data into a hash, which is a fixed-size value. Hash functions are often used to check the accuracy of data because a hash value will change if the data is changed [33].

Along with these encryption techniques, there are other protocols and standards that can be used to protect bioinformatics data. Using the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols, for example, data sent over the internet is frequently encrypted. The Health Insurance Portability and Accountability Act (HIPAA), a US law, establishes standards for the security of medical data, including bioinformatics data [34]. Protecting sensitive data requires HIPAA compliance, which calls for the implementation of suitable security measures like encryption.

In the research [33], a cryptographic and bioinformatics-based encryption and decryption technique is offered. In the suggested algorithm, a novel technique using RNA and deoxyribonucleic acid as keys is produced for safe data encryption and decryption procedures over a communication to conceal message from intruder or third party.

There are many different libraries that can be used for data encryption in bioinformatics. These libraries are designed for different programming languages and support different encryption algorithms. Examples could be:

- *pycrypto*: It is a library [35] designed for the Python language and includes various encryption algorithms. For example, it supports algorithms like AES, RSA, DES.
- *PyNaCl*: It is a library [36] designed for the Python language and includes modern encryption algorithms. For example, it supports algorithms like Curve25519, Salsa20, Poly1305.
- *Cryptography*: It is a library [37] designed for the Python language and includes various encryption algorithms. For example, it supports algorithms like AES, RSA, DES.
- *Bouncy Castle*: It is a library [38] designed for the Java language and includes various encryption algorithms. For example, it supports algorithms like AES, RSA, DES.
- OpenSSL: It is a library [39] designed for the C language and includes various encryption algorithms. For example, it supports algorithms like AES, RSA, DES.
- Crypto++: It is a library [40] designed for the C++ language and includes various encryption algorithms. For example, it supports algorithms like AES, RSA, DES.

These examples are just a few libraries you can use for data encryption, there are actually many more libraries available and each optimized for different programming languages and different encryption algorithms. Which library to use depends on factors such as the programming language you use and the encryption algorithm you need.

D. Role-based Access Control (RBAC)

RBAC is a technique for controlling access to computer or network resources based on the responsibilities of certain individuals within an organization [11]. To limit the activities users may perform inside a system, administrators can establish roles for users and give rights to those roles using RBAC. For instance, a system administrator may designate a "customer support" role with access to see and change client data but not to remove it. Users with the "customer support" position would then have the necessary access to carry out their job responsibilities, but they would not be able to make modifications that would jeopardize the security or integrity

of the system. Access to resources and information may be managed effectively and flexibly with RBAC.

With role-based access control (RBAC), roles are allocated to users to decide what activities they may do on a system. For instance, in an accounting system, accountants could be able to issue invoices and manage payments but not carry out additional tasks like managing databases.

Role-Based Access Control (RBAC) is a commonly used method for controlling access to resources in a computer system, and there are many libraries and tools that can be used to implement the RBAC method for bioinformatics. These may include:

- *py-rbac*: For bioinformatics applications, this Python module offers a straightforward and adaptable framework for role-based access management [41].
- *BioPerl RBAC*: This gives users access to files, directories, and other resources in addition to a collection of Perl modules and an RBAC implementation for bioinformatics applications [42].
- *BioJava RBAC*: RBAC may be easily and adaptably implemented in web-based bioinformatics applications with the help of this Java package [43].

E. Differential Privacy Techniques

Differential privacy [12], [13], [44] is a statistical and data science concept that aims to secure the privacy of people whose data is being gathered and examined. It's a mathematical framework for calculating the level of privacy in a dataset and ensuring that analysis of the data doesn't adversely impact the privacy of the people whose data is included. In addition, it can be counted as a sub-topic of data anonymization techniques.

There are a number of different algorithms [45], [46], [47] that can be used to achieve differential privacy, each with its own strengths and weaknesses. Some of the most commonly used algorithms include:

- *Randomized response:* To hide the values of individual records, this algorithm [46] involves introducing random noise to the data. It is envisioned that the additional noise will make it challenging to determine the actual value of any given record while still enabling accurate analysis of the entire dataset.
- *Laplace mechanism:* This algorithm [45, 46] is similar to randomized response, but it introduces noise into the data that is based on the Laplace distribution rather than random noise. By doing this, it is assured that the extra noise is distributed in a way that protects the privacy of the people whose data is used.
- *Exponential mechanism:* This algorithm [45, 46] is a more advanced version of the Laplace mechanism, which takes into account the sensitivity of the data, as well as the desired level of privacy. It ensures that the added noise is distributed in a way that maximizes the privacy of the individuals whose data is included.
- *k-anonymity*: This algorithm [47] divides the data into "clusters" of k records, each of which has k records or more. This makes it difficult to link particular data to particular individuals because it guarantees that every individual record is a part of a group of at least k records.
- *Gaussian:* This algorithm [45] adds noise to the data based on the Gaussian distribution. The amount of noise added is determined by the standard deviation, which is chosen based on the desired level of privacy.

These algorithms can be used in a variety of different settings, including in surveys, medical research, and social media data analysis. While each algorithm has its own strengths and weaknesses, they are all designed to help protect the privacy of individuals whose data is being collected and analyzed.

IV. Advantage Of Known Data Security Techniques In Bioinformatics

Data security techniques in bioinformatics have many advantages. Some of these are:

- *Ensures the security of your data*: Your data is safeguarded against unwanted access, alteration, or destruction thanks to data security procedures. By doing this, you can retain the accuracy and integrity of your data while also ensuring its security.
- *Protects the privacy of your data:* By using data security strategies, you can keep your data private and prevent others from viewing it. This protects both the privacy of your data and your personal life.
- *Prevents data loss*: Data security measures guarantee that your data is backed up and prevent data loss. This enables you to recover your data in the event that it is lost or corrupted.
- *Reduces operating costs*: Data security methods lower operating expenses as well as the costs associated with data loss or data security breaches.
- *Increases customer trust*: Techniques for data security boost consumer confidence and allay customers' worries about your data. This enhances the reputation of your company and fosters more consumer loyalty.

V. Applications Of Known Data Security Techniques

A. Dataset

The sample dataset [48] is related to lung-cancer disease. There are 1000 patient records suffering from lung cancer disease. The dataset includes 25 columns such as patient age, smoking, level of lung cancer etc. All the attributes and values of the dataset are shown in Table I.

B. Applications

a) Data anonymization: There is an example of a simple function that takes a dataset as input and replaces certain identifiable information with placeholder values:

```
#Data anonymization example with the patients suffering lung cancer disease
def data_anonymization(data):
    # Replace patient age with placeholder values
    data['age'] = 0

    # Replace gender with placeholder values
    data['gender'] = 'Unknown'

    # Replace patient genetic risk with placeholder values
    data['genetic_risk'] = '-1'

    # Replace any sensitive health information with placeholder values
    data['lung_cancer_level'] = 'level'
    data['alcohol_use'] = 'alcohol use'
    data['smoking'] = 'smoking'

    return data
```

This function replaces patient age, gender, genetic risk, and any sensitive health information in the dataset with placeholder values.

TABLE I. DATASET ATTRIBUTES

Column	Value
Patient ID	Numeric
Age	14-73
Gender	0: Male, 1: Female
Air Pollution	1-8
Alcohol Use	1-8
Dust Allergy	1-8
Occupational Hazard	1-8
Genetic Risk	1-8
Chronic Lung Disease	1-8
Balanced Diet	1-8
Obesity	1-8
Smoking	1-8
Passive Smoker	1-8
Chest Pain	1-8
Coughing of Blood	1-8
Fatigue	1-8
Weight Loss	1-8
Shortness of Breath	1-8
Wheezing	1-8
Swallowing Difficulty	1-8
Clubbing of Finger Nails	1-8
Frequent Cold	1-8
Dry Cough	1-8
Snoring	1-8
Lung Cancer Level	Low, Medium, High

b) *Data masking:* There are a few libraries in Python that can be used to mask data in a bioinformatics dataset, such as pandas and NumPy. Here's an example of a function that masks the data in a bioinformatics dataset by replacing certain columns with random values:

```
#Data masking example with the patients suffering lung cancer disease
import pandas as pd
import numpy as np

def mask_bioinformatics_data(filepath, columns_to_mask):
    # read the dataset into a pandas DataFrame
    df = pd.read_csv(filepath)

    # replace the values in the specified columns with random values
    for column in columns_to_mask:
        if column in df.columns:
            df[column] = np.random.randint(1, 100, size=len(df))
    # now the columns have been changed with random values
    return df
```

This function can be used by providing the path to the bioinformatics dataset and a list of columns that you want to mask.

c) *Data Encryption:* One way to encrypt data in Python is to use the cryptography library. Here is an example of a function that encrypts the data in a bioinformatics dataset using the Advanced Encryption Standard (AES) algorithm:

```
#Data encryption example with the patients suffering lung cancer disease
from cryptography.fernet import Fernet

def encrypt_bioinformatics_data(filepath, columns_to_encrypt, key):
    # read the dataset into a pandas DataFrame
    df = pd.read_csv(filepath)
    # encrypt the specified columns
    for column in columns_to_encrypt:
        if column in df.columns:
            # create a Fernet object using the key
            cipher = Fernet(key)
            # encrypt each value in the column and update the DataFrame
            df[column] = df[column].apply(lambda x:
                                          cipher.encrypt(str(x).encode()))
    return df
```

This function can be used by providing the path to the bioinformatics dataset and a list of columns that you want to encrypt, and also a key should be provided that is used for the encryption. A key can be generated using the Fernet.generate_key() method which returns a URL-safe base64 encoded key. This key should be a secret and should be stored at a secure place, also it is significant to be careful when transferring the key to other parties.

d) Role-based Access Control (RBAC): In Python, the Flask-RBAC library can be used to implement RBAC in a web application that uses the Flask framework. Here is an example of a function that sets up RBAC for a bioinformatics dataset:

```
#Role based access control example with the patients having lung cancer disease
from flask_rbac import RBAC

def setup_rbac(app, roles, permissions):
    rbac = RBAC(app)

    # create roles
    for role in roles:
        rbac.create_role(name=role)

    # create permissions
    for permission in permissions:
        rbac.create_permission(name=permission)

    # assign permissions to roles
    rbac.set_role_permissions(roles['bio_admin'],
                              [permissions['view_patientdata'],
                               permissions['edit_patientdata'],
                               permissions['delete_patientdata']])
    rbac.set_role_permissions(roles['bio_patient'],
                              [permissions['view_patientdata']])
    rbac.set_role_permissions(roles['bio_guest'], [])
```

This function can be used in the flask applications by passing the flask app instance and list of roles it can be created along with the permission that each role will have.

The RBAC object can be used to create roles and permissions and to assign permissions to roles. In this example,there are three roles bio_admin, bio_patient and guest, and three permissions view_patientdata, edit_patientdata, and delete_patientdata. The bio_admin role has all the permissions, the bio_patient role has the view_patientdata permission, and the guest role has no permissions.

The RBAC object can be used in the views to protect routes and check patient permissions,

```
@app.route('/data', methods=['GET'])
@rbac.allow(['bio_admin', 'bio_patient'], methods=['GET'])
def view_patientdata():
    # code to view patient data
```

In this example, only the bio_patient with bio_admin and patient role will be able to see the data. This is just a simple example of how RBAC can be set up for a bioinformatics dataset using the Flask-RBAC library. In a real-world scenario, this function can be customized to suit the specific needs of the applications, such as adding or removing roles and permissions, and integrating it with the authentication and authorization system.

e) Differential Privacy Techniques: Lapcian, Gaussian, and Exponential methods are built in the Python programming language. After the construction of the algorithms, “Age” feature is selected from the dataset [48] to be anonymized with the help of these algorithms. In conclusion, these algorithms are compared with some comparison parameters such as privacy guarantee, accuracy, noise, sensitivity, and robustness.

The Laplacian function applied in this study is as follows.

```python
import numpy as np

def laplacian_mechanism(data, epsilon, sensitivity):
    '''
    Add Laplacian noise to a bioinformatics dataset for differential privacy.
    data: The bioinformatics data to be protected
    epsilon: The privacy budget parameter
    sensitivity: Sensitivity of the data.
    return: The data with added Laplacian noise
    '''
    scale = sensitivity / epsilon
    noise = np.random.laplace(loc=0, scale=scale, size=data.shape)
    return data + noise
```

The Gaussian function applied in this study is as follows.

```python
import numpy as np

def gaussian_mechanism(data, epsilon, sensitivity):
    '''
    Gaussian mechanism for differential privacy.
    data: Data to be privatized.
    epsilon: Privacy budget (epsilon).
    sensitivity: Sensitivity of the data.
    return: Privatized data.
    '''
    # Scale factor
    scale = sensitivity / epsilon
    # Add noise sampled from a Gaussian distribution
    privatized_data = data + np.random.normal(scale=scale, size=data.shape)
    return privatized_data
```

The Exponential function applied in this study is as follows.

```python
import numpy as np

def exponential_mechanism(data, epsilon, sensitivity, utility_function):
    '''
    Exponential mechanism for differential privacy.
    data: Data to be privatized.
    epsilon: Privacy budget (epsilon).
    sensitivity: Sensitivity of the data.
    utility_function: Utility function to evaluate the quality of the privatized data.
    return: Privatized data.
    '''
    scale = sensitivity / epsilon
    num_elements = data.shape[0]
    probabilities = np.exp(utility_function(data) / scale) / np.sum(np.exp(utility_function(data) / scale))
    privatized_data = np.random.choice(data, size=num_elements, p=probabilities)
    return privatized_data
```

The above algorithms are compared using criteria including privacy guarantee, accuracy, noise, sensitivity, and robustness. In this part of the study, *NumPy* [49] library is used to apply the comparison parameters. The details of these comparison parameters are:

- *Privacy guarantee:* Compare how much information about specific data points is revealed by each method. The method with the lowest information leak would be considered the best in terms of ensuring privacy.
- *Accuracy:* Compare how accurately each technique's results were produced. The most accurate method would be the one that yields the best results in terms of accuracy.
- *Noise addition:* Compare the amount of noise added to the data by each technique. Regarding noise addition, the method that introduces the least amount of noise would be deemed to be the best.
- *Sensitivity:* Compare how much of a change in the data each technique will be able to detect. In terms of sensitivity, the most sensitive method would be regarded as the best.
- *Robustness:* Compare the ability of each technique to maintain privacy guarantees when the data distribution is unknown or varies. The technique that is the most robust would be considered the best in terms of robustness.

In the following table, the decimal values of each comparison parameter are shown after applying the above algorithms for "Age" attribute.

TABLE 2. COMPARISON TABLE OF DIFFERENTIAL PRIVACY TECHNIQUES

Privacy Technique	Privacy Guarantee	Accuracy	Noise	Sensitivity	Robustness
Laplacian	9.6041	9.6041	-0.5029	30.0	10.8223
Gaussian	8.0728	8.0728	0.3407	30.0	8.1036
Exponential	10.2430	10.2430	10.2430	30.0	9.7274

According to the results given in Table 2, Fig 1 represents the graph of the results.

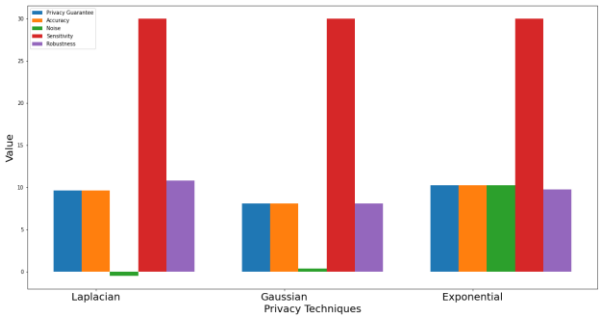

Fig. 1 Graph of the comparison parameters such as privacy guarantee, accuracy, sensitivity, and robustness

The results of analysis of each algorithms are given as follows:

- *Privacy guarantee:* The results you obtained indicate how well each technique is able to protect the privacy of the data while preserving its utility. The privacy guarantee measures how much privacy is preserved by each technique, where a higher value means that the data is more protected. The Gaussian technique has a lower privacy guarantee than the Laplacian and Exponential techniques, which means that it provides less privacy.
- *Accuracy:* The accuracy measures how well the data is preserved after applying the technique, where a higher value means that the data is more accurate. The Laplacian and Exponential techniques have a higher accuracy than the Gaussian technique, which means that they preserve more of the original data.
- *Noise addition:* The noise measures how much noise has been added to the data, where a higher value means that more noise has been added. The Laplacian technique has added less noise than the Gaussian and Exponential techniques, which means that it preserves more of the original data.
- *Sensitivity:* The sensitivity measures how sensitive the data is to the addition of noise, where a higher value means that the data is more sensitive. The three techniques have the same sensitivity, which means that the data is equally sensitive to the addition of noise.
- *Robustness:* The robustness measures how much the data has changed as a result of applying the technique, where a higher value means that the data has changed more. The Laplacian technique has a higher robustness than the Gaussian technique, but the Exponential has higher robustness than Laplacian.

Based on these results, it can be concluded that the Laplacian technique provides the best balance between privacy and utility, as it provides the highest privacy guarantee and accuracy, and the lowest noise and robustness. However, the choice of technique will depend on the specific requirements of the use case and the trade-off between privacy and utility that is acceptable.

VI. RESULT

In this study, five known data security techniques in bioinformatics were examined and some libraries related to data security are explained. With the help of some of these libraries, some security functions such as data_anonymization(), mask_bioinformatics_data(), encrypt_bioinformatics_data(), and setup_rbac() were builded in Python programming language. These methods were applied for a dataset [48] including 1000 patients who have lung cancer disease.

In order to prevent re-identification of individuals, data anonymization removes personal identifiers from a dataset, such as age, gender, genetic risk, etc. Personal identifiers are typically either removed entirely from the dataset or replaced with placeholder values to accomplish this. Data anonymization aims to render re-identification of individuals based on the remaining data extremely challenging, if not impossible.

While personal identifiers can still be used for analysis or research, the risk of re-identification is significantly reduced when they are "masked" or obscured. Personal identifiers are typically replaced with random values or symbols, such as the asterisk (*), to perform masking. Masking aims to strike a balance between the need for data access and the need to safeguard individual privacy.

It is difficult to compare the results of these two techniques because it depends on the dataset, the required level of security, the particular identifiable information, and other factors. In terms of privacy protection, data anonymization is regarded as being more effective than masking because it completely removes personally identifiable information. However, because some of the data may be lost or replaced with placeholder values, anonymization can also reduce the dataset's usefulness for particular research or analysis purposes. However, masking can still offer a degree of privacy protection while allowing the data to be used in more ways.

In addition to other data security techniques, Laplacian, Gaussian, and Exponential algorithms which are some of the differential privacy techniques are applied to the lung-cancer dataset [48]. As a sample feature, “Age” attribute is anonymized with these algorithms. As a result of the comparison parameters such as privacy guarantee, accuracy, noise, sensitivity, and robustness, it can be concluded that The Laplacian technique offers the highest privacy guarantee (9.6041) and accuracy (9.6041), the lowest noise (-0.5029) and robustness (10.8223), and the best balance between privacy and utility. However, the technique chosen will depend on the particular needs of the use case and the acceptable trade-off between privacy and utility.

It's crucial to remember that neither anonymization nor masking by themselves can always guarantee privacy protection. To further prevent unauthorized access to the data, it is crucial to implement additional security measures such as access controls and monitoring even after using one of these techniques. For future work, with the advancement of data security techniques, new security models can be constructed in addition to the mentioned techniques.

VII. REFERENCES

[1] Y. Sakakibara, “Grammatical Inference in Bioinformatics”, IEEE Transactions on Pattern Analysıs and Machine Intelligence, Vol. 27, No. 7, July 2005.

[2] L. S. Heath , N. Ramakrishnan, “The emerging landscape of bioinformatics software systems”, Computer, https://doi.org/10.1109/mc.2002.1016900, 2002.

[3] D. Kaloudas, N. Pavlova, R. Penchovsky, “EBWS: Essential Bioinformatics Web Services for Sequence Analyses”, IEEE/ACM Transactions on Computational Biology and Bioinformatics, Vol. 16, No. 3, 2019.

[4] N. Rapin, C. Kesmir, S. Frankild, M. Nielsen, C. Lundegaard, S. Brunak, O. Lund, “Modelling the Human Immune System by Combining Bioinformatics and Systems Biology Approaches”, Journal of Biological Physics, 32: 335–353, 2006.

[5] M. Thomas, A. Daemen, B. DeMoor, “Maximum Likelihood Estimation of GEVD: Applications in Bioinformatics”. IEEE/ACM Transactions on Computational Biology and Bioinformatics, Vol. 11, No. 4, 2014.

[6] M. Armstrong, J. Thomas, B. Henson, A. Kirby, M. Galloway, “Bioinformatics Cloud Security”, 2019 IEEE Cloud Summit. https://doi.org/10.1109/cloudsummit47114.2019.00018, 2019.

[7] A. Tamersoy, G. Loukides, M. ErcanNergiz, Y. Saygin, B. Malin, “Anonymization of Longitudinal Electronic Medical Records”, Vol. 16, No. 3, 2012.

[8] G. Loukides, A. Gkoulalas-Divanis, “Utility-Aware Anonymization of Diagnosis Codes”, IEEE Journal of Biomedical and Health Informatics, Vol. 17, No. 1, 2013.

[9] J.-X. WEI, M.-H. LIU, Z.-Q. LU, J. Wang, S. CHEN, Y. LAN, G.-Z. FENG, “Minimization of masking in signal detection from Chinese spontaneous reporting databases based on data removal strategy”, 2020 13th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), IEEE, 2020.

[10] N. Dowlin, R. Gilad-Bachrach, K. Laine, K. Lauter, M. Naehrig, J. Wernsing, “Manual for Using Homomorphic Encryption for Bioinformatics”, Proceedings of the IEEE, https://doi.org/10.1109/jproc.2016.2622218, 2017.

[11] D. Shin, G-J. Ahn, J. S. Park, “An application of directory service markup language (DSML) for role-based access control (RBAC)”, Proceedings 26th Annual International Computer Software and Applications, 2002.

[12] C. Dwork, “Differential privacy,” in Proc. 33rd Int. Colloquium on Automata, pp. 1–12, 2006.

[13] J. L. Raisaro *et al*., "Protecting Privacy and Security of Genomic Data in i2b2 with Homomorphic Encryption and Differential Privacy," in *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 15, no. 5, pp. 1413-1426, 1 Sept.-Oct. 2018.

[14] J. M. Struble, P. Handke, R. T. Gill, “Genome Sequence Databases: Genomic, Construction of Libraries”, Editor(s): Moselio Schaechter, Encyclopedia of Microbiology (Third Edition), Academic Press, Pages 185-195, ISBN 9780123739445, 2009.

[15] A. Schmidt, I. Forne, A. Imhof, “Bioinformatic analysis of proteomics data”, BMC Syst Biol 8 (Suppl 2), S3, 2014.

[16] M. D. Sorani, J. K. Yue, S. Sharma, G. T. Manley, A. R. Ferguson, “Genetic data sharing and privacy. Neuroinformatics”, doi: 10.1007/s12021-014-9248-z. PMID: 25326433; PMCID: PMC5718357, 2015 Jan;13(1):1-6.

[17] T. King, L. Brankovic, P. Gillard, “Perspectives of Australian adults about protecting the privacy of their health information in statistical databases”, International Journal of Medical Informatics, 81(4):279–289, 2012.

[18] J. Gauthier, A. T. Vincent, S. J. Charette, N. Derome, “A brief history of bioinformatics, Briefings in Bioinformatics”, Volume 20, Issue 6, Pages 1981–1996, 2019.

[19] R. A. Irizarry, B. Hobbs, F. Collin, Y. D. Beazer-Barclay, K. J. Antonellis,, U. Scherf, T. P. Speed, “Exploration, normalization, and summaries of high density oligonucleotide array probe level data”, Biostatistics, Apr;4(2):249-64, 2003.

[20] A. Harbola, D. Negi, M. Manchanda, R. K. Kesharwani, “Bioinformatics and biological data mining”, Editor(s): Dev Bukhsh Singh, Rajesh Kumar Pathak, Bioinformatics, Chapter 27, Pages 457-471, ISBN 9780323897754., Academic Press, 2022.

[21] E. Abrahams, G. S. Ginsburg, M. Silver, “The Personalized Medicine Coalition: goals and strategies”, Am J Pharmacogenomics., 5(6):345-55, 2005.

[22] H. Han, W. Liu, “The coming era of artificial intelligence in biological data science”, BMC Bioinformatics 2019, 20(Suppl 22):712, China. 22-24, June 2019.

[23] M. Krassowski, V. Das, S. K. Sahu, B. B. Misra, “State of the Field in Multi-Omics Research: From Computational Needs to Data Mining and Sharing” Front Genet, 10;11:610798, 2020.

[24] R. Lowe, N. Shirley, M. Bleackley, S. Dolan, T. Shafee, “Transcriptomics technologies”, Plos Computational Biology, 13(5), 2017.

[25] D. Goodman, C. O. Johnson, D. Bowen, M. Smith, L. Wenzel, K. Edwards, “De-identified genomic data sharing: the research participant perspective”, Journal of community genetics, 8(3), 173–181, 2017.

[26] https://pypi.org/project/pynonymizer/, Pyanonymizer, 11.01.2023.

[27] https://pypi.org/project/Faker/0.7.4/, Faker, 11.01.2023.

[28] https://pypi.org/project/anon/, anon 0.0.1, 10.01.2023.

[29]https://data2knowledge.atlassian.net/wiki/spaces/DSDEV/pages/12943436/Session+2+SSCM+DataSHIELD+tutorial, SSCM DataSHIELD tutorial, 10.01.2023.

[30] https://amnesia.openaire.eu/, High accuracy Data Anonymization, 10.01.2023.

[31] https://techdocs.broadcom.com/us/en/ca-enterprise-software/devops/test-data-management/4-9/getting-started/getting-started-with-fast-data-masker.html, Getting Started with Fast Data Masker, 10.01.2023.

[32] https://www.datarobot.com/, DataRobot, 10.01.2023.

[33] V. Siddaramappa, K. B. Ramesh, "Cryptography and bioinformatics techniques for secure information transmission over insecure channels," 2015 International Conference on Applied and Theoretical Computing and Communication Technology, Davangere, pp. 137-139, India, 2015.

[34] https://www.hhs.gov/hipaa/for-professionals/privacy/index.html, The HIPAA Privacy Rule, 11.01.2023.

[35] https://pypi.org/project/pycrypto/, pycrypto 2.6, 10.01.2023.

[36] https://pypi.org/project/PyNaCl/, PyNaCl 1.5.0, 10.01.2023.

[37] https://pypi.org/project/cryptography/, cryptography 39.0.0, 10.01.2023.

[38] https://www.baeldung.com/java-bouncy-castle, Introduction to BouncyCastle with Java, 10.01.2023.

[39] https://wiki.openssl.org/index.php/Compilation_and_Installation, OpenSSL, 10.01.2023.

[40] https://www.cryptopp.com/, Crypto++® Library 8.7, 10.01.2023.

[41] https://pypi.org/project/py-rbac/, py-rbac 20.12.3, 10.01.2023.

[42] https://bioperl.org/, BioPerl, 10.01.2023.

[43] https://biojava.org/, BioJava , 10.01.2023.

[44] A. Dyda, M. Purcell, S. Curtis, E. Field, P. Pillai, K. Ricardo, H. Weng, J. C. Moore, M. Hewett, G. Williams, C. L. Lau, “Differential privacy for public health data: An innovative tool to optimize information sharing while protecting data confidentiality”, Patterns, Volume 2, Issue 12, 2021.

[45] M. U. Hassan, M. H. Rehmani, J. Chen, “Differential Privacy Techniques for Cyber Physical Systems: A Survey”, IEEE Communications Surveys & Tutorials, Vol. 22, No. 1, First Quarter 2020.

[46] K. C. Gadepally, S. Mangalampalli, “Effects of Noise on Machine Learning Algorithms Using Local Differential Privacy Techniques”, 2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), Conference Paper, 2021.

[47] P. Samarati, L. Sweeney, “Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression”, Computer Science, 1998.

[48] https://data.world/cancerdatahp/lung-cancer-data/workspace/file?filename=cancer+patient+level%20data+sets.xlsx. , Lung Cancer Data, 10.01.2023.

[49] https://numpy.org/, NumPy library, 10.01.2023.