SUMMARY

In recent years, machine learning has increased in popularity, driven by advancements in computing power and the exponential growth of data. The availability of large datasets is crucial for deep learning solutions, significantly enhancing model performance. In healthcare, increased computing power has empowered researchers to utilize machine learning models to analyze and extract features from various medical datasets effectively. The results extracted from data using these trained machine learning models can contribute to personalized medicine by enabling tailored treatment plans, thereby improving patient outcomes. However, healthcare data is typically private and confined to specific legal entities, creating isolated data islands. Different hospitals, for example, possess electronic health records (EHR) of distinct patient groups, and sharing these records is challenging due to their sensitive nature. This restriction hinders for example, the development and implementation of deep learning approaches that require extensive healthcare datasets to train accurate, high-quality models. Furthermore, the fragmented nature of healthcare data prevents comprehensive analysis across diverse patient populations, limiting the ability to derive insights that could lead to better diagnostic tools and treatments. As a result, there is a pressing need for methods that can leverage distributed data without compromising privacy and security.

In this paper, we present a study on the application of Federated Learning (FL) for extracting mutational signatures from genomic data in healthcare. Traditional methods of analyzing mutational signatures, which are unique patterns in cancer genomes indicating different mutational processes, often require centralized data storage. This poses significant privacy concerns, as healthcare data is highly sensitive and typically decentralized across various institutions. Despite the genomic data on human cancer being available as anonymized versions of the raw data, future research may want to train and extract signatures on raw data which is governed by privacy regulations such as GDPR.

The primary problem addressed in our research is the challenge of extracting meaningful mutational signatures from decentralized genomic data while ensuring data privacy. Current centralized approaches necessitate data aggregation, which is infeasible due to privacy and legal constraints. Here, we explore the use of FL, a technique where machine learning models are trained across multiple decentralized devices holding local data samples, without exchanging the data itself. Specifically, we utilize Non-negative Matrix Factorization (NMF) and autoencoders (AE) as methods for mutational signature extraction within the FL system. The study aims to evaluate the performance of these methods compared to traditional centralized approaches. This involves testing the FL system on both synthetic datasets and real-world genomic data, including those from prominent cancer genome repositories such as The Cancer Genome Atlas (TCGA) and the International Cancer Genome Consortium (ICGC). We assess the accuracy of the methods in a FL setting in identifying mutational signatures and compare it with centralized methods, considering the trade-offs in computational time due to the distributed nature of FL.

Our main findings demonstrates that FL achieves comparable accuracy to centralized approaches in extracting mutational signatures. However, it incurs higher computational costs, highlighting a trade-off between data privacy and computational efficiency. Despite this, FL presents a viable solution for privacy-preserving analysis in genomic studies, ensuring that sensitive patient data remains decentralized. Future work is suggested to focus on optimizing computational resources and improving the efficiency of FL algorithms, particularly in handling large-scale genomic datasets. Additionally, exploring adaptive techniques for local hyperparameter tuning and aggregation of models trained on differently sized datasets could further enhance the performance and applicability of FL in healthcare.

In conclusion, we demonstrate that FL can be effectively applied to mutational signature extraction, offering a promising approach for privacy-preserving, collaborative genomic research. Our results support the notion that FL can bridge the gap between the need for large, diverse datasets and the imperative of protecting patient privacy.

Federated Learning for Mutational Signature Extraction in Healthcare

June 2024

Frederik Rasmussen
Aalborg University
frasm19@student.aau.dk

Kevin Risgaard Sinding
Aalborg University
ksindi19@student.aau.dk

ABSTRACT

Cancer is a genetic disease caused by various factors, with each mutational process leaving a unique, identifiable signature within the genome. These mutational signatures provide valuable insights into the origins and development of cancer, aiding in the creation of targeted treatments. This study evaluates the use of federated learning (FL) for mutational signature extraction using Non-negative Matrix Factorization (NMF) and autoencoders (AE). The framework assesses performance on both synthetic and real-world genomic datasets, comparing FL methods to centralized approaches. The results show that FL achieves comparable accuracy in identifying mutational signatures but incurs increased computational time due to the distributed nature of the process. This suggests that FL is a viable alternative for privacy-preserving analysis, though it requires careful management of computational resources.

KEYWORDS

machine learning; federated learning; privacy preservation; mutational signature extraction; non-negative matrix factorization; autoencoders

1 INTRODUCTION

In recent years, machine learning has risen in popularity, driven by advancements in computing power and the rapid growth of data [12]. The large amount of available data is essential for deep learning solutions, greatly affecting the performance of the models.

In the field of healthcare, the increase in computing power has enabled researchers and scientists to leverage machine learning models to effectively analyze and extract features from various medical datasets. The outcome of trained models in personalized medicine contributes to tailored treatment plans, leading to improved patient outcomes [17]. However, healthcare data is typically private and isolated to legal entities, resulting in *data islands* in which the data is not allowed to be shared due to privacy restrictions. For instance, different hospitals possess electronic health records (EHR) of distinct patient groups, and sharing these records is challenging due to the sensitive nature of the data. This restriction impedes for example, the development and application of deep learning approaches that rely on large healthcare datasets to train accurate and high-quality models.

To address this issue, federated learning has emerged as a potential solution. Federated learning (FL) is a way of training machine learning algorithms collaboratively by distributing the algorithm to the data, instead of the data to the algorithm [29]. This approach mitigates the privacy concerns associated with traditional approaches, as raw data remains decentralized and is processed at the institutions.

In cancer genomic studies, mutational signature analysis stands out as an important task [33]. This analysis involves the examination of patterns in genetic mutations within cancer genomes, providing insights into the underlying mutational processes driving the development of a cancer. By leveraging methods and machine learning techniques, such as NMF [11] and AE [25], researchers can efficiently analyze large-scale genomic datasets to uncover these mutational signatures [36]. Traditionally, mutational signature extraction approaches are performed on genomic datasets that are derived from a large collection of cancer patient samples. These datasets are compiled through collaborative efforts involving genomic initiatives such as The Cancer Genome Atlas (TCGA) [32] and the International Cancer Genome Consortium (ICGC) [7]. The Pan-Cancer Analysis of Whole Genomes (PCAWG) study [8] offers a publicly available and extensive repository of cancer genome datasets, encompassing over 2,600 whole genomes sourced from the ICGC. These datasets represent a diverse collection of cancer patient samples, capturing various mutation types suitable for mutational signature extraction.

It is important to note that while TCGA and ICGC repositories contain vast amounts of data, our study specifically focuses on utilizing the anonymized subset provided by the PCAWG initiative. This subset maintains a high level of data aggregation, thereby safeguarding patient privacy.

While mutational signature extraction methods such as NMF and AE have traditionally been applied in centralized settings, their adaptation to distributed settings present unique challenges. The increasing adoption of privacy restrictions and data-sharing constraints by legal entities possessing cancer patient data necessitates the exploration of alternative methodologies. Although current datasets for mutational signature analysis are anonymized and comply with privacy regulations, it is imperative to consider the potential challenges that may arise in the future.

As legislation evolves and data collection practices advance, there is a possibility that mutational signature analysis could encounter privacy issues. For instance, future analyses may involve the integration of additional sensitive data or require access to more comprehensive datasets. In such scenarios, maintaining patient privacy while conducting mutational signature extraction in a distributed setting becomes increasingly complex.

Existing research in FL has demonstrated its efficiency in various applications, including predictive maintenance, natural language processing, and healthcare analytics [24, 31]. For instance, Google has successfully applied FL to improve predictive text suggestions on mobile keyboards without compromising user privacy [22]. Similarly, researchers have explored FL in medical imaging tasks, demonstrating its potential to develop robust models while preserving patient privacy [35]. As the size of the datasets increases in the future, researchers may face the challenge of efficiently analyzing distributed data.

While existing research in the landscape of cancer genomics has explored the application of NMF and autoencoders for mutational signature extraction in a centralized setting, there is limited research on extending these methods to a decentralized setting. FL presents unique challenges and opportunities for mutational signature extraction, including privacy-preserving model training across distributed data sources and collaboration between institutions. However, the adaptation of NMF and autoencoder-based approaches to FL frameworks for mutational signature extraction remains largely unexplored.

The challenge lies in developing or leveraging novel methods and models that can effectively integrate NMF and autoencoder-based methods into a federated setting while addressing the complexities of decentralized data sources. Additionally, ensuring the scalability and efficiency of these techniques in federated settings pose a significant challenge when dealing with large-scale genomic datasets.

Therefore, the objective of this project is to explore the performance of a federated learning system utilizing NMF and AE as mutational signature extraction methods compared to centralized learning approaches.

Our main findings demonstrate the potential of federated learning as a privacy-preserving approach for collaborative model training across decentralized data sources. By extending mutational signature extraction methods to a federated learning setting, we can extract mutational signatures at a level that is competitive in performance to in a centralized learning setting.

2 BACKGROUND

Before delving into our research, it is important to understand two key concepts; *federated learning* and *mutational signatures*. These concepts serve as the foundation for exploring the performance of a federated learning system using mutational signature extraction methods. In addition, we explain the methods associated with mutational signature extraction.

2.1 Federated Learning

Federated learning is an approach to collectively train a model across different devices such as phones or computers (referred to as *clients*) without sharing their private data. The learning task is coordinated by a central *server* that handles the construction and maintenance of a *global* model to the clients in the environment. Each client has a local training dataset that is isolated from the central server and other clients, therefore they can be seen as data islands. The central server distributes a copy of the global model (local model) to each client, where the client performs local computations, updates the local model, and sends this back to the central server. The global model is updated and maintained by the central server alone, and therefore only the updated local model needs to be communicated from a client. In addition, the local models are discarded once they have been sent back to the central server as they serve no purpose once they have been applied. The process of updating the global model is typically done by an *aggregation* algorithm, wherein the server integrates the model updates received from the clients to refine the global model [24]. Equation 1 shows the FedAvg [24] algorithm for aggregating model updates. This aggregation mechanism is crucial for ensuring that the global model reflects the collective training of the shared models. Here, W represents the global model or parameters that are being updated through aggregation, W_i represents the model parameters of the ith local model or client and n represents the total number of local models or clients participating in the federated learning process.

$$W = \frac{1}{n}\sum_{i=1}^{n} W_i \tag{1}$$

In contrast to centralized learning, where the model and training data reside at the same location, one major benefit of federated learning is that it separates model training from requiring direct access to the original training data. In this way, FL can potentially overcome privacy and security concerns with training models on data that is sensitive of nature. Since the clients only communicate the model updates to the central server, the privacy of the data at the clients remains confidential in the environment [24].

The learning task [24] in a federated setting can be described as the following global objective function:

$$f(w) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(w) \quad \text{where} \quad F_k(w) = \frac{1}{n_k} \sum_{i \in P_k} f_i(w) \tag{2}$$

where w represents the model parameters, K is the number of clients participating in the federated learning process, n is the total number of data samples across all clients, and n_k is the number of data samples on client k. Each client k independently computes its local objective function $F_k(w)$, which is the average of the individual loss functions $f_i(w)$ over its local dataset P_k. The global objective function $F(w)$ is then a weighted sum of these local objective functions, with weights proportional to the number of data samples n_k on each client.

In a conventional federated learning setting, the data already reside at each client. However, it is possible to create a federated learning setting based on data partitioning where a dataset is partitioned and distributed to different clients. Horizontal FL and vertical FL are two approaches that address different data partitioning scenarios.

In *horizontal* federated learning, each client has a dataset with the same set of features but different samples. This means that clients share a common feature space but the samples are distributed across clients. Conversely, in *vertical* federated learning, each client has a dataset with different features but the same set of samples. This means that clients share a common sample space but have different attributes or features for these samples [34].

2.2 Mutational Signatures

Mutational signatures are unique patterns of mutation types that arise from different mutational processes [5]. These processes can include environmental exposures, endogenous cellular processes, or DNA repair mechanisms. Each mutational process leaves a characteristic imprint on the genome, resulting in a unique mutational signature [10]. These signatures are typically represented as matrices or vectors, capturing the frequency of specific mutation types across different contexts. For example, one mutational signature might indicate exposure to ultraviolet (UV) radiation from the sun, while another might suggest a malfunction in DNA repair mechanisms.

In this study, we focus on single base substitutions (SBSs), which involve changes to individual nucleotide bases, and exclude other types of mutations such as double base substitutions (DBSs) and small insertions or deletions (indels). SBSs are a class of *somatic mutations* commonly sorted into six subtypes; C:G > A:T, C:G > G:C, C:G > T:A, T:A > A:T, T:A > C:G, and T:A > G:C. Strand symmetry ensures that changes in one strand maintain the same alterations in the paired strand, preserving the structural integrity of the double helix. For instance, a change from C to T on one strand corresponds to a change from G to A on the complementary strand [28].

The surrounding sequence context of a mutating base is critical in defining the mutation's signature. By examining the nucleotide bases neighboring the mutating base, one on each side, a 4×4 matrix of potential nucleotide combinations can be generated. Considering the six subtypes of mutations, this yields a 96-dimensional feature vector for each mutational signature. Aggregating these feature vectors across multiple samples results in a $96 \times N$ matrix, where N represents the number of samples in the dataset, forming the *mutational catalog* of the dataset [28].

To extract mutational signatures from genomic data, various methods are employed. This includes techniques such as NMF and AE. NMF can be used to decompose the genomic data into a set of basis signatures and their corresponding activities in each sample. AE, a type of neural network, can also be used to learn representations of the data and extract mutational signatures. These methods will be elaborated further in the upcoming sections.

2.2.1 COSMIC database. The COSMIC Mutational Signatures database [5] contains signatures derived from extensive analysis of the PCAWG dataset and curated scientific papers. These signatures are identified through thorough examination, by experts in the field, of specific exposures, providing a comprehensive representation of mutational processes across various cancer types.

2.3 Non-negative Matrix Factorization

Non-negative matrix factorization (NMF) is a matrix decomposition technique with the constraint that all entries must be non-negative. It decomposes an original data matrix $A \in \mathbb{R}^{m \times n}$ into two sub-matrices $W \in \mathbb{R}^{m \times k}$ and $H \in \mathbb{R}^{k \times n}$ such that $A \approx WH$.

Mathematically, NMF can be formulated as an optimization problem, where F denotes the Frobenius norm:

$$\min_{W,H} \quad \|A - WH^T\|_F$$

The goal of the approximation is to minimize the reconstruction error between the original data matrix A and the product of the two matrices W and H, using loss functions such as Mean Squared Error (MSE) and Kullback-Leibler divergence (KL). One approach is to use the Frobenius norm, which measures the difference between A and WH as the square root of the sum of the squares of all elements. To achieve this, an iterative process is used where the initial values of W and H are adjusted to bring the product closer to A continuing until the convergence requirement is met.

As a result, NMF provides two matrices: the basis matrix W, which contains the basis vectors or components (signatures), and the coefficient matrix H, which contains the weights or coefficients that linearly combine the basis vectors to approximate the original data. The basis matrix W represents the underlying features or building blocks of the data, while the coefficient matrix H indicates how these features combine to form the original data. This decomposition is particularly useful for revealing latent structures and patterns in the data.

2.4 Autoencoders

Autoencoders are a type of neural network architecture aimed at reconstructing its original input. The main objective is to learn an informative representation of the data. More formally, the goal of an autoencoder is to learn two functions $A : \mathbb{R}^n \to \mathbb{R}^p$ (encoder) and $B : \mathbb{R}^p \to \mathbb{R}^n$ (decoder), where n represents the dimension of the input space and p is the dimension of the latent space. The encoder takes the input data and reduces it to a lower-dimensional latent space representation. This process involves compressing the input data into a more compact form by extracting important features. The decoder receives the latent space representation from the encoder and attempts to reconstruct the original input data from this representation. Together, the encoder and decoder is trained to minimize the reconstruction error between the input data and the reconstructed output [14]. The reconstruction error is typically represented by a loss function $\mathcal{L}$ that measures the dissimilarity between the original input and reconstructed output. This training process encourages the autoencoder to learn a compressed version of the input data that captures its characteristics.

3 RELATED WORK

In this section, we review the existing literature on the application of NMF and AE in a federated learning setting, focusing on their relevance to decentralized data analysis and collaborative model training.

3.1 Distributed NMF

In their paper, Qian et al. [27] proposed a novel distributed NMF algorithm, DSANLS, tailored for federated environments. Their method distributes the NMF computation across multiple nodes (K) in a federated setting.

Initially, the size of the input matrix is reduced by using matrix sketching, where a smaller, approximate representation of the original data matrix is created. Here each row indices $\{1, 2, ...m\}$ of the input matrix is partitioned into K disjoint sets $I_1, I_2, ...I_K$ where $I_r \subset \{1, 2, \ldots, m\}$ is assigned to node r. Similarly each column indices $\{1, 2, ...n\}$ is partitioned into disjoint sets $J_1, J_2, ...J_K$. To achieve load balance, the data is split evenly over the nodes $|I_r| \approx \frac{m}{K}$ and $|J_r| \approx \frac{n}{K}$ for each node. The factor matrices $WandH$ are also assigned to nodes i.e. node r stores and updates W_{I_r} and H_{J_r} reducing the size of each non-negative least squares (NLS) subproblem.

Each node then computes NMF on its subset of the data. This decentralized approach ensures that the sensitive data remains local to each node, thereby enhancing data privacy and security.

Subsequently, these partial computations are aggregated by combining the W_i and H_i matrices to reconstruct the global factor matrix, thus preserving data privacy and security.

Their findings indicate that DSANLS outperforms state-of-the-art NMF algorithms like MU [23], HALS [20], and ANLS/BPP [21] in terms of accuracy, as measured by relative error over time, showcasing its effectiveness in a distributed environment.

3.2 Distributed AE

A study by Cha et al. [16] proposed a method utilizing overcomplete autoencoders (e.g., the hidden layer having a higher dimension than the input layer) for generating latent representations of original data in the context of vertical federated learning.

Patient	Features			
	f_1	f_2	$\cdots$	f_n
p_1	🟦	🟦	🟥	🟩
p_2	🟦	🟦	🟥	🟩
$\vdots$	🟦	🟦	🟥	🟩
p_m	🟦	🟦	🟥	🟩

Figure 1: Features ($f_1, f_2, ..., f_n$) and patients ($p_1, p_2, ..., p_m$) where each colored column represents the partitioned data at hospital A, hospital B, and hospital C, respectively

They vertically divide a dataset into subsets and train two overcomplete autoencoder models; one in a centralized setting on the partitioned data and another in a federated setting on the generated latent representations. In each subset, a local overcomplete autoencoder model was trained to generate latent representations of the original data. These latent representations can be considered anonymized, as they are distinct from the original data and do not contain identifiable information, thus respecting privacy restrictions. Subsequently, the latent representations were aggregated for further model training by concatenating the latent data from each site as shown below.

$$D' = [D'_i|D'_{\mathrm{i}+1}|\cdots|D'_k]$$

where D' represents the aggregated latent representations and D'_i represents the latent representation from a local autoencoder on partition D_i.

The accuracy of the trained models using the original data are compared to the trained models using aggregated latent data were compared then, respectively.

Three distinct datasets (Adult income [13], Schwannoma [15], and eICU [26] datasets) were vertically partitioned into subsets to simulate distributed data settings. The Adult income dataset and Schwannoma dataset were split into 3 partitions (or sites) each and the eICU was split into 7 partitions. The comparison of the performance between the centralized model and the federated model revealed that there was a minimal loss in performance across the Adult income, Schwannoma, and eICU datasets, respectively. The comparison highlights the effectiveness of federated learning with autoencoders in preserving privacy while achieving competitive results compared to centralized learning.

Our FL system will compare the performance of mutational signature extraction methods in a federated learning setting against centralized learning setting.

4 METHODOLOGY

In this section, we propose our approach for implementing mutational signature extraction methods in a federated learning system. This approach aims to leverage the distributed nature of federated learning to explore the performance of mutational signature extraction methods in a decentralized setting as opposed to a traditional centralized setting.

The proposed pipeline consists of several key components, each playing a crucial role in the implementation and evaluation of the federated learning system. The pipeline can be summarized as follows:
Step 0: The central server initializes the global model parameters, setting the stage for the federated learning process.
Step 1: The central server sends a copy of the global model to the clients.
Step 2: Each client independently trains the received model on its local data, extracting mutational signatures through specified methods.
Step 3: Clients communicate their locally updated models back to the central server.
Step 4: The server aggregates these updates using the FedAvg algorithm, updating the global model.
These steps are repeated for multiple rounds to progressively improve the global model.

The pipeline ensures that data privacy is maintained by keeping the raw data decentralized at the clients while leveraging federated learning to collaboratively train a robust global model. This methodology is designed to facilitate the evaluation of mutational signature extraction methods in a federated setting.

In the following sections, we will describe each step of the proposed framework in detail, focusing on the federated learning setting, the extraction methods used, the generation of synthetic datasets for validation, and the clustering techniques applied to refine the results. Additionally, we will explain the implementation details of the framework, including the use of the Flower framework to support the federated learning environment, and the procedures for evaluating and outputting the results.

4.1 Federated Learning Setting

We create a horizontal federated learning setting by partitioning a dataset by samples (genomes) into a number of distinct subsets. Each client in the setting will have its own subset to perform a mutational signature extraction method on. This is done to simulate the idea of data islands where a dataset of a client cannot be shared with other clients. To emulate varying scenarios, the dataset is divided into three subsets (simulating local data for three clients), both evenly and unevenly distributed among the clients.

In this setting, we have a central server that coordinates interactions among K clients. The central server is responsible for managing the results received from clients and updating the global model using a strategy. The strategy specifies how results from the different clients are aggregated. For this, we use the FedAvg [19] strategy. In addition, the server manages the number of rounds that clients undergo in the federated learning process. Here, a round refers to the complete cycle of communication and computation that occurs between the central server and the participating clients. We describe procedure of the central server in Algorithm 1.

Algorithm 1 Central Server Federated Learning

```
Input: Number of rounds $R$
Output: None
Initialize global model parameters $\theta$
for each round in range($R$) do
    Initialize empty list to store client updates $G \leftarrow []$
    for each client $k$ in range($K$) do
        Send global model parameters $w$ to client $k$
    end for
    Wait for all clients to complete their local computations
    for each client $k$ in range($K$) do
        Receive local model update $w_k$ from client $k$
        Append $w_i$ to $G$
    end for
    Update global model: $w = FedAvg(G)$
end for
```

4.2 Extraction Methods

4.2.1 Federated NMF. Building upon the NMF algorithm's execution in Section 2.3 in our distributed setup, NMF computations occur independently at each client using distinct input datasets. This local computation yields a separate W_i matrix at each client, representing the components extracted from its respective dataset. These W_i matrices are then communicated to the central server.

Upon receiving all W_i matrices from the distributed clients, the server performs aggregation by computing the average of the W_i matrices:

$$W = \frac{1}{n} \sum_{i=1}^{n} W_i$$

where n is the number of clients. This aggregation step combines the component information derived from different datasets, creating a unified representation of the deconstructed features across all the diverse datasets processed by the clients. After the first round of running NMF, the global W matrix is given to the NMF client as an input. Then, the global aggregated W matrix is used as an input into the NMF algorithm, and the reconstruction error between the W and H matrices is iteratively minimized at each client based on the original input dataset during each round.

4.2.2 Federated AE. The AE implementation in this study is based on the DeepMS model proposed by Pei et al. [25], adapted for a federated learning setting. It consists of an encoder-decoder neural network architecture which reduces the dimensionality of the genomic dataset. We describe the procedure of the federated AE in Algorithm 3.

Algorithm 2 Federated NMF Client

```
Input: Mutational catalog $A$, Number of components $k$, Initial
mutational signature matrix $W_i$ (optional)
Output: Updated mutational signature matrix $W$, Loss
Initialize matrix $A$ (mutational catalog) and components $k$
if $W_i == \emptyset$ then
    $W, H$ = NMF($A$, $k$)
else
    $W, H$ = NMF($A$, $W_i$)
end if
Loss = reconstruction error($A, W, H$)
return $W$ and Loss to the server
```

Algorithm 3 Federated AE

```
Input: Mutation frequency matrix $M$
Output: Locally updated model $\theta'$
Initialize local model parameters $\theta$
Update local model with global model parameters
for run in range(10) do
    Train model and obtain $W$, $H$, and loss $l$
end for
Perform clustering on $W$
for each signature in $W$ do
    for component in $C$ do
        Apply KMeans clustering with component clusters
        Find cluster centroids, silhouette score, and inertia score
    end for
end for
Compute auxiliary loss and select the best cluster centroids
Set signatures to the optimal cluster centroids
Update local model: $\theta \longrightarrow \theta'$
return locally updated model $\theta'$
```

In the federated learning setup, the clients in the environment collaboratively train the global model on their local dataset. Each client initializes the model with the global parameters θ received from the central server and performs local training. The local training process involves multiple epochs of forward and backward passes through the AE. During each epoch, the training data is passed through the encoder to generate latent representations, which are then fed to the decoder to reconstruct the original data.

Once the AE has been trained, we cluster the resultant signatures, and optimize the clustering results to identify the final mutational signatures. We elaborate on this approach in Section 4.3.1. In addition, once all clients have finished their local training and signature extraction, the resulting locally updated model parameters are communicated back to the central server from each client.

4.3 Synthetic Dataset Generation

A synthetic dataset is generated to facilitate the validation of mutational signatures in a controlled environment. This synthetic dataset is based on mutational signatures from the COSMIC database. The process for generating the synthetic dataset includes the following steps:

To proceed, two matrices need to be created: a signature matrix and a signature exposure matrix. The signature matrix will consist of the COSMIC signatures that form the basis of the generated samples. The signature exposure matrix will contain information about the number of mutations each mutational signature from the COSMIC database should contribute to the given sample.

Signature selection. A predefined set of mutational signatures is read from an input file containing all COSMIC signatures, from which a specified number of signatures are randomly selected. To minimize redundancy and ensure diversity among the selected signatures, the selection process ensures that the cosine similarity between any pair of selected signatures does not exceed a pair-wise cosine similarity of < 0.7.

Signature exposure matrix. The process of constructing the signature exposure matrix involves several steps. For each sample, a random value between 500 and 1000 is generated to represent the total number of mutations. This range is chosen to reflect a realistic number of mutations typically observed in biological samples. These mutations are then distributed across the respective signatures in the matrix. Additionally, two signatures are randomly chosen to have an exposure of 0 for each sample. This decision is based on the understanding that not all mutational signatures are active in every sample. By setting two signatures to zero, we simulate the presence of inactive signatures, which is a common scenario in real-world mutational data. The dot product of the signature exposure matrix and the selected signature matrix is calculated, resulting in a matrix representing the mutational profile of each sample based on its exposure to each signature. Finally, Poisson noise is added to the dot product matrix as a percentage of each signature's mutation amount [28].

The synthetic dataset enables the validation of mutational signatures under controlled conditions by comparing the identified signatures against the mutational signatures used to create the dataset. This method ensures that the synthetic data closely simulates real-world mutational profiles. The procedure of generating a synthetic dataset is shown in Algorithm 4.

The synthetic dataset is designed to access the performance of the different methods in a controlled environment.

4.3.1 Clustering.

To achieve an accurate representation of the results from the AE and NMF, post-processing using clustering is necessary. NMF is run multiple times because the results can vary with each computation, indicating that a single computation might not adequately represent NMF's performance. The same logic applies to AE, as the latent representation may not be clear from a single run.

K-means. To cluster the data, we use the K-means [30] algorithm to partition the input data into K clusters. It assigns each data point to the nearest cluster centroid and iteratively updates the centroids until convergence. This method effectively minimizes within-cluster variance.

Algorithm 4 SynthData

```
Input: Array of 5 random COSMIC signatures $S$, Number of samples $N$
Output: Mutation catalog
for $i = 1$ to $N$ do
    Generate random vector $x$ by shuffling $[1, 1, 0, 0, 1]$
    Apply Poisson distribution to each element in $x$
    Compute the sum $xs$ of all elements in $x$
    Generate probability vector $P$ as $P = (x/xs) \times \text{Uniform}(500, 1000)$
    Compute mutational profile $V$ as the element-wise product of $S$ and $P$
    Generate noise array $Noise$ with 96 elements from Poisson distribution
    Compute the sum $NoiseSum$ of all elements in $Noise$
    Normalize $Noise$ as $Noise = (Noise/NoiseSum) \times P_i \times 0.01$
    Compute the final mutational profile $Final$ as the element-wise sum of $V$ and $Noise$
end for
```

NMF. To achieve an accurate representation of the outcomes, the NMF algorithm is executed several times for each number of components. The results are then combined and used as input to the K-Means algorithm. The number of clusters that K-means are looking for is set to the component value used to create the input. K-means then fits the given components into the original component value clusters. This creates an instance of K-means for each different set of components used. Here, a limit of 300 iterations is set. To calculate the optimal amount of components in the original dataset, the silhouette score is determined for each instance of K-Means. This silhouette score is then used in an auxiliary loss function, along with the reconstruction error of the NMF. These values are normalized to get them on a similar scale, contributing an equal amount to the calculation. Equation 3 is used to measure the optimal amount of clusters.

$$auxiliary = \widehat{loss} - 1 * \widehat{silhouette} \tag{3}$$

Following [28], we use the lowest value obtained from the auxiliary loss function to determine the optimal amount of clusters, as it maximizes the silhouette score and minimizes the reconstruction loss. An example of the loss function compared to each component value is showcased in Figure 2. This example is produced based on the federated NMF, using three rounds, with an input of a synthetic dataset, created using five signatures, evenly distributed among three clients. After the optimal amount of clusters is found, the cluster centroids are extracted, which are the mutational signatures.

The SigProfilerAssignment [18] tool is used to extract the weights (exposure). The tool is extracting the weight by locking the W and A matrix and fitting the weight in the $A \approx WH$ equation.

AE. When the AE is done training, it extracts 200 latent variables. These latents are used as an input for the K-means algorithm. Then K-means is run with a K value ranging from 2-200, which represent the number of latents. For each instance of K-means, the silhouette score is determined along with the inertia score. The inertia is a way of measuring how well the data points are clustered, specifically the sum of the squared distances between each data point and the nearest cluster center. The inertia score and silhouette score are used to create the auxiliary loss as described in Equation 4. Instead of using the reconstruction error, the inertia score is used as the

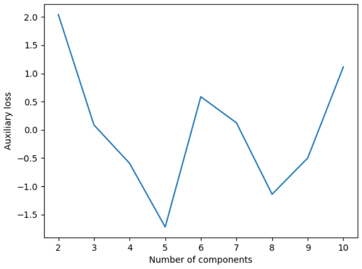

Figure 2: Example of an auxiliary loss function with the component value 5

first parameter. The reasoning behind using the inertia score is that the reconstruction loss for the autoencoder does not change post training. If the inertia score was not added to the equation it would only consider the silhouette score, meaning that it would favour less clusters. The optimal amount of clusters is found using the same approach as 2. The cluster centroids are found using the determined K, resulting in the final mutational signatures for the AE. The weight for the AE is determined using the SigProfilerAssignment [18] tool, equal to the approach with NMF.

$$auxiliary = \widehat{inertia} - 1 * \widehat{silhouette} \tag{4}$$

4.4 Implementation

This section will provide a comprehensive explanation of the implementation of the FL system and the integration of new extraction methods into the Fl system.

4.4.1 Flower. To facilitate the federated learning environment, we have utilized the Flower framework [3]. Flower provides a platform for federated learning, providing the necessary architecture and implementation to distribute and train a shared model over a set of participating clients. By leveraging Flower, we are able to extend the mutational signature methods using built-in functions that allow us to distribute the results of models being trained and the aggregation of this.

4.4.2 Signature Extraction. The architecture of the framework is designed to facilitate the evaluation of various extraction methods. To initiate the extraction process, the user specifies both the dataset and the signature extraction method. Each extraction method contains information about its clustering approach, which varies depending on whether the method is NMF or an autoencoder. When adding a new method, the only prerequisite is that the input and output formats remain consistent throughout the pipeline. Specifically, the input to the extraction methods includes the dataset, the search space for the latent variables when using an autoencoder, and the components when using NMF. The output generated by these methods includes signatures, error metrics, and weights.

4.4.3 Evaluating the Signatures. Before beginning the evaluation process, it is important to determine whether a ground truth set of signatures has been provided. If a ground truth set is available, the evaluation will be conducted against these provided signatures, which were used to create the synthetic dataset. If no ground truth set of signatures is available, the evaluation will instead compare the identified signatures against the COSMIC signatures database.

4.4.4 Outputting the Results. When the evaluation is completed, files containing the results from the specified extraction methods are produced. The results include information about the number of extracted signatures, the matched signatures and pair-wise cosine similarity, above a certain threshold.

5 EXPERIMENTAL EVALUATION

In this section, we explain how we perform experiments with mutational signature extraction methods on both synthetic and real datasets and evaluate their performance in a federated setting compared to in a centralized setting.

5.1 Experimental Setup

We outline the setup for our experiments aimed at evaluating the performance of a mutational signature method. We utilize synthetic datasets as well as a real dataset to assess the method's performance under different circumstances.

5.1.1 Datasets and Horizontal Division of Data. The evaluation of the mutational signature extraction methods is performed on two different types of datasets; a synthetic dataset and a real dataset referred to as WGS PCAWG dataset [8]. In addition, for the experiments we want to run the federated learning setting with three clients to simulate three distinct data islands. The experiments on the methods in the centralized learning setting will be performed on the original dataset, whereas in the federated learning setting the methods will be run on the partitions based on the original dataset. We present the following datasets for our experiments in Table 1.

Table 1: Dataset Information

Dataset	Type	Division	Dimension
Synth5	Synthetic	3 clients	96×500
WGS	Real	3 clients	96×2780

Synth5

The Synth5 dataset is a synthetic genomic dataset generated using the procedure shown in Algorithm 4. It consists of 96 mutation types (rows) and includes 500 samples (columns). It is based on 5 random signatures from the COSMIC database [5] which do not exceed a pair-wise cosine similarity of < 0.7. We horizontally divide this into three partitions to simulate three data islands.

WGS

The WGS dataset [8] is a real genomic dataset consisting of 2780 columns representing the occurrence or frequency of mutations.

5.1.2 Method Evaluation. Following [28], we differentiate between the accuracy when applying the methods to real and synthetic datasets. We employ cosine similarity to compare the extracted signatures with the known signatures as shown in Equation 5.

$$\text{cosine similarity}(\mathbf{A}, \mathbf{B}) = \frac{\mathbf{A} \cdot \mathbf{B}}{|\mathbf{A}| \times |\mathbf{B}|} \tag{5}$$

where $\mathbf{A}$ and $\mathbf{B}$ represent the vectors of extracted and known signatures, respectively. This measure helps quantify the similarity between the signatures, providing insight into the accuracy of the extraction methods.

Real Dataset. We validate extracted signatures against known COSMIC [6] signatures. Here, we employ the *linear sum assignment* [9] algorithm to optimally match the extracted signatures with known COSMIC signatures based on cosine similarity. The algorithm maximizes the total cosine similarity by assigning each extracted signature to a unique known signature. We choose to employ a threshold of 0.80 for cosine similarity of the extracted signatures because it presents are more stringent criterion. Extracting signatures from real-world datasets can be considered more challenging compared to synthetic datasets, where we have control over the signatures present in the data. This is because we cannot be certain that an extracted signature from a real dataset actually represents a distinct mutational signature.

Synthetic Dataset. We validate extracted signatures with known (predefined) signatures used during dataset generation. The procedure of matching the extracted signatures with known signatures is identical to when dealing with real datasets, but with the addition of comparing the estimated weights from the synthetic dataset with the true weights. This is done using measures such as mean squared error (MSE), mean absolute error (MAE), and root mean squared error (RMSE). We quantify matches with cosine similarity similarly to extracted signatures in real datasets. In this controlled environment of a synthetic dataset, where the signatures are precisely defined and the dataset generation process is known, we have the advantage of assessing the performance of the signature extraction algorithm with a high level of confidence. Here, we employ a threshold of 0.95 for cosine similarity as this underscores the need for a high degree of similarity between the extracted signatures and the known signatures to consider them valid matches. We want this threshold to reflect the expectation of a nearly perfect match, given the controlled nature of the synthetic dataset.

5.1.3 Metrics.

a) ***Accuracy*** Accuracy is crucial for evaluating the performance of mutational signature extraction methods. It measures the fidelity of the extracted signatures compared to the ground truth or validated signatures. In the context of our experiments, accuracy can be defined as the percentage of correctly identified signatures against predefined (synthetic) or validated (real) signatures. We use the cosine similarity thresholds previously described to classify a signature as a correctly identified signature. For this, we express the accuracy as proposed in Equation 6.

$$\text{accuracy} = \frac{\text{number of correctly identified signatures}}{\text{total number of actual signatures}} \tag{6}$$

b) ***Efficiency*** In our evaluation, we consider the execution time of each method. In particular, we are interested in the efficiency of each method executing in a federated setting compared to the centralized setting.

6 RESULTS

This section contains the result from the different extraction method along with a detailed analysis of the evaluation.

6.1 Synthetic Dataset

The performance of the methods in identifying mutational signatures was evaluated using a synthetic dataset for both settings. Table 2 summarizes the results of this evaluation.

Method	Loss	CT	Found	>0.8	>0.95	b>0.95	b>0.99
NMF	MSE	27.93	8	8	7	5	2
NMF	KL	43.64	10	8	7	5	4
AE	MSE	265.45	65	54	12	4	0
AE	KL	260.39	63	44	6	4	0
FedNMF	MSE	37.78	7	6	4	4	1
FedNMF	KL	52.41	8	7	4	4	1
FedAE	MSE	599.67	51	39	10	4	1
FedAE	KL	589.11	43	29	11	4	2

Table 2: Results for Synth5 (true K = 5)

6.1.1 Centralized setting.

NMF results. NMF was able to identify all five signatures when using a cosine similarity threshold of 0.95 with both the KL divergence and MSE loss functions, giving it and accuracy of 1. However, a closer examination reveals that KL divergence loss function is more effective in accurately identifying the signatures. Specifically, with a higher cosine similarity threshold of 0.99, NMF was able to find four signatures using KL divergence as the loss function, whereas it was able to find two using the MSE loss function.

AE results. The AE was able to find four signatures in the synthetic dataset given the five signatures in total, resulting in an accuracy of 0.80. When considering the quality of the identified signatures based on the cosine similarity threshold we sat on 0.95, the AE produce similar results across the two employed loss functions. It finds a considerable number of signatures with cosine similarity above 0.8, indicating reasonable similarity to known signatures. However, this number drops significantly when considering the more strict threshold of 0.95. This suggests that while the AE can identify signatures, they may not always closely match the known signatures.

6.1.2 Federated Setting with Even Partition (3 Clients).

NMF results. Federated NMF, with an even partitioning of the dataset, was able to identify four out of five signatures using a cosine similarity threshold of 0.95 with both the KL divergence and MSE loss functions, giving it an accuracy of 0.8. However, when the cosine similarity threshold was increased to 0.99, federated NMF identified one signature with both KL divergence and MSE loss functions.

A notable difference between the two loss functions is the computation time. NMF with KL divergence as the loss function required 15 seconds longer to complete the computations compared to when using MSE as the loss function.

AE results. Our experiments compared the performance of the DeepMS AE trained in a federated learning setting against a centralized learning setting. Specifically, we evaluated the ability of both setups to identify mutational signatures following the approach in Section 4.2. We found that the AE was able to identify the same number of mutational signatures in both federated and centralized settings. The experimental results in Table 2 showed that autoencoder model did not suffer a loss in terms of accurately extracting mutational signatures. Similarly to the results in the centralized setting, the federated AE found four signatures out of the five represented in total, resulting in an accuracy of 0.80. This indicates that the federated learning approach does not compromise the accuracy of the method on a synthetic dataset.

However, we observed that the federated learning setting suffered from an additional computation time compared to the centralized setting. For instance, in the centralised learning setting, the computation times (in seconds) for the AE (MSE) and AE (KL) models were 265.45 and 260.39. In contrast, the FL approach incurred higher computation times, with AE (MSE) and AE(KL) models requiring 599.67 and 589.11 seconds, respectively. This was primarily due to the communication overhead involved in the federated learning process. The process of federated learning requires multiple rounds of communication and computation to collaboratively train a model across multiple clients while preserving data privacy. These iterative rounds amplify the computation time in the federated learning setting. It should also be noted that all the local autoencoders in the federated learning setting train and extract signatures over a total of three rounds to reap the benefit of a shared model trained collaboratively over the set of clients.

Method	Loss	CT	Found	>0.8	>0.95	b>0.95	b>0.99
FedNMF	MSE	40,27	5	4	1	1	0
FedNMF	KL	43,50	9	5	2	2	0

Table 3: Results for Synth5 (true K = 5)

6.1.3 Federated Setting with 60-20-20 Partition (3 Clients).

NMF results. Federated NMF with a 60-20-20 partition of the dataset identified 1 signature using the MSE loss function and 2 signatures using the KL divergence loss function with a cosine similarity threshold of 0.95. In this case, the computation times for both loss functions were nearly identical, differing by approximately three seconds.

When the cosine similarity threshold was increased to 0.99, neither method was capable of finding matching signatures.

AE results. Unfortunately, as part of performing the experiments with the federated AE we encountered an issue on aggregating local models trained on local datasets of different dimensions. This led to the central server being unable to perform the aggregation as intended and send the updated global model to the clients for the next round. These results would have been ideal to examine the impact of local models trained on different sized datasets on the global model. Based on these, we would be able to see for each round how well local models train and extract signatures compared to when local models were trained on local datasets of equal size. We discuss this more in Section 9.

6.2 Real-World dataset

We evaluated the performance of the two extraction methods in both settings on the WGS PCAWG dataset. Table 4 summarizes the results of this evaluation.

Method	Loss	CT	Found	>0.8	>0.95
NMF	MSE	4323.50	77	8	0
NMF	KL	4116.64	86	9	0
AE	MSE	463,61	24	0	0
AE	KL	354,06	9	0	0
FedNMF	MSE	4458.77	95	5	0
FedNMF	KL	4278.34	98	6	0
FedAE	MSE	1202.47	53	1	0
FedAE	KL	942.12	5	0	0

Table 4: Results for WGS PCAWG dataset

6.2.1 Centralized setting.

NMF results. The centralized NMF on real data determined a component value of 77 when emloying the MSE loss function and 86 for when employing the KL divergence loss function. However, only 8 signatures for MSE and 9 signatures had a cosine similarity of 0.8.

AE results. In the centralized learning setting, the findings showed that the AE with MSE loss identified 24 preliminary signatures, none of which exhibited a cosine similarity above the designated threshold of 0.8. Similarly, the AE with KL loss identified 9 preliminary signatures, also with none surpassing the cosine similarity threshold of 0.8. These findings suggest that while the AE is able to identify a set of preliminary signatures, it did not achieve high cosine similarity scores with the current threshold.

When examining the extracted signatures, it is noteworthy that the highest cosine similarity match observed was the cosmic signature SBS34 [2], with a cosine similarity of 0.66 using the MSE loss function and a cosine similarity of 0.65 using the KL divergence loss function. Conversely, the lowest cosine similarity match was the cosmic signature SBS17b [1], with a similarity score of 0.203.

NMF results. Federated NMF on a real dataset with even partitioning determined a component value of 95, indicating the identification of 95 signatures. However, five of these 95 signatures had a cosine similarity of 0.8.

AE results. When running the federated AE on the WGS PCAWG dataset, we found that the choice of loss function had a significant impact on the result. Here, when running the AE with the MSE loss function it extracted a single signature above the designated cosine similarity threshold of 0.80. Compared to its counterpart in the centralized setting, the results show an increase in performance in terms of accuracy in a federated setting.

Throughout the training of the federated AE we noticed that the training loss was not improving when using the KL divergence loss method. The loss would remain stable regardless of number of runs for locally training the federated AE and despite getting the benefit of receiving the global model. This could be connected to the significant difference in number of extracted signatures between using MSE loss function and KL divergence loss function.

Centralized	Loss	CT	Found	>0.8	>0.95
NMF	MSE	5184.58	95	5	0
NMF	KL	4785.32	98	6	0

Table 5: Summation of results, real dataset, uneven partitioning

6.2.2 Federated Setting with 60-20-20 Partition (3 Clients).

NMF results. Federated NMF on a real dataset with an uneven partitioning determined a component value of 95, indicating the identification of 95 signatures. However, only 5 of these 95 signatures had a cosine similarity of 0.8.

6.3 Summary of Findings

While the federated methods maintains comparable accuracy to the centralized methods in identifying mutational signatures from a synthetic dataset, they incur a higher computation time due to the overhead associated with federated learning. This demonstrates the trade-off between computation time and data privacy when training a shared model on data not accessible in a centralized setting. This could be a key consideration when choosing between centralized and federated approaches for mutational signature extraction.

6.3.1 Comparison of Methods. To effectively compare the methods for extracting mutational signatures, we assessed their performance using different component ranges and loss functions on both synthetic and real-world datasets.

Component range for real vs. synthetic data. In the experiments, the component range varied between the synthetic and real datasets. For the synthetic dataset, we tested component ranges from two to ten, given the controlled nature and known complexity of the dataset (true K = five). However, for the real-world WGS PCAWG dataset, the component range was extended to 2 to 100 to capture the potentially higher complexity and variability inherent in real genomic data.

NMF vs. AE.

- **NMF (Non-negative Matrix Factorization):**
  - *Synthetic Data:* NMF demonstrated robust performance in the centralized setting, accurately identifying the majority of signatures with high cosine similarity (e.g., five out of five signatures with KL and MSE loss). In the federated setting, the accuracy slightly decreased but remained competitive.
  - *Real Data:* NMF identified a large number of preliminary signatures, with nine signatures for KL and eight for MSE meeting the cosine similarity threshold of 0.8. Federated NMF provided comparable results, but was able to find fewer signatures above the threshold.
- **AE (Autoencoder):**
  - *Synthetic Data:* AE identified a large number of preliminary signatures but struggled to achieve high cosine similarity matches. This pattern persisted across both centralized and federated settings.
  - *Real Data:* AE's performance was lower compared to NMF, with no signatures exceeding the 0.8 cosine similarity threshold. The federated AE showed similar trends but required more computational resources due to communication overhead.

Federated vs. Centralized Approaches. While centralized approaches generally provided higher accuracy, federated methods demonstrated significant potential, particularly for NMF. The federated setting introduced some loss in performance, likely due to data partitioning and communication overhead, but it also offered advantages in scenarios where data sharing is restricted. The federated AE faced challenges in model aggregation, especially with varying dataset sizes, which highlighted the need for further research in optimizing federated learning frameworks for such tasks.

6.3.2 Impact of Data Partitioning. We analyzed the performance under two partitioning strategies: even partition (3 clients) and 60-20-20 partition (3 clients).

Even Partitioning. With an even partitioning strategy, each client received an equal portion of the dataset. This approach generally led to better performance in federated learning, as evidenced by the federated NMF:

- **Federated NMF:** Even partitioning allowed for more balanced training across clients, resulting in higher accuracy and more signatures meeting the cosine similarity thresholds.

Uneven Partitioning (60-20-20). The 60-20-20 partitioning strategy, where one client received a larger portion of the dataset than the others, highlighted the challenges of imbalanced data distribution:

- **Federated NMF:** The accuracy decreased under this partitioning strategy. The imbalance likely led to less effective training for clients with smaller data portions, impacting the overall model performance. In the synthetic dataset, this strategy resulted in fewer high-similarity signatures.

Key Observations.

- **Balanced Data Distribution:** Ensuring an even distribution of data across clients is crucial for maximizing the effectiveness of FL methods. Balanced data partitioning leads to more uniform model updates and better overall performance.
- **Communication Overhead:** FL introduces additional computation time due to communication overhead between clients and the central server. This was evident in the increased computation times for federated AE and NMF.
- **Signature consistency:** Out of the five signatures identified in the FL setting with both even and uneven partitioning on real data, three of these signatures were also found in the centralized version with a cosine similarity threshold of 0.8. This overlap suggests that federated learning can achieve comparable accuracy to centralized methods in identifying

key mutational signatures, despite the challenges posed by data partitioning.

7 DISCUSSION

In this study, we research the challenge of adapting NMF and autoencoder-based methods for mutational signature extraction to a FL setting, addressing the complexities of decentralized data sources. We aimed to evaluate the performance of a FL system employing NMF and AE in comparison to traditional centralized approaches.

Our research builds upon existing methodologies by extending them to federated settings, where data privacy and decentralization are paramount concerns. While previous studies have primarily focused on centralized learning approaches, our work explores the potential of federated learning in genomic data analysis.

The strength of our approach lies in its integration of NMF and autoencoder-based methods into a federated learning system, offering a privacy-preserving alternative to centralized approaches. However, challenges remain in ensuring the scalability and efficiency of these techniques in federated settings, particularly when dealing with extensive genomic datasets. Additionally, the adaptation of NMF and autoencoders to decentralized environments requires careful consideration of data distribution and communication overhead.

7.1 Poorly Trained Local Models

One of the notable challenges we observed was that the performance of the global model could suffer from the aggregation of a poorly trained local model. When all of the clients finished their local training, a potential scenario was that a locally trained model would perform worse than the global model. There was no mechanism for handling poorly trained local models resulting in a worse global model being distributed for the next round to the clients. Addressing this challenge is important for achieving efficient training of global models in federated learning settings.

To mitigate the impact of poorly trained local models, several approaches can be researched. Implementing an adaptive weighting mechanism, where the contribution of each locally trained model to the global model is based on its performance. For instance, local models that achieve higher accuracy or lower loss on a validation set can be assigned greater weight during aggregation, ensuring that well-performing models have a more significant impact on the global model. In the case of our proposed framework, this would involve implementing a custom strategy on the central server to handle this aggregation.

7.2 Hyperparameter Tuning in Federated Learning

In FL, hyperparameter tuning presents a unique challenge due to the decentralized nature of training data and computation. Unlike centralized settings where the entire dataset is available for training of the model, FL involves training models across a network of clients, each with its own local dataset and computation resources. Each client operates under stringent privacy constraints, limiting the frequency and the scope of data access. Consequently, hyperparameter tuning algorithms must be able to overcome the challenge of optimizing model performance with minimal data availability. For this, it would be necessary to determine where in the setting to optimize the hyperparameters and how.

Hyperparameter tuning in FL can be approached by optimizing both local training and global aggregation parameters. The objective would be to minimize the over loss across all clients by adjusting the hyperparameters such as learning rates, the number of local training epochs, and aggregation strategies. For this project, we employ a fixed set of hyperparameters for the local models. These hyperparameters include a learning rate of $1e{-}3$, 500 local training epochs, a batch size of 8 and a latent dimension of 200. This is based on the SEEF [28] framework for extracting mutational signatures in a centralized setting. Here, the hyperparameters were based on a combination of anecdotal observations and the use of the tool Optuna [4].

7.3 Cosine Similarity Threshold

During the testing of the federated approach against the centralized approach, it was observed that both settings identified the correct number of components and signatures when using the best signatures with a cosine similarity greater than 0.95. This indicates that the federated approach is capable of finding the same signatures as the centralized approach. However, due to the rounds of aggregation in the federated setting, achieving the > 0.99 cosine similarity threshold becomes more challenging.

When reviewing other studies that use cosine similarity to measure mutational signatures, values between 0.80 [25] and 0.95 [11] are commonly reported. This suggests that the federated approach, while slightly less accurate than the centralized approach, performs comparably well given the same amount of data.

When dealing with real data, it is challenging to achieve a cosine similarity threshold of 0.95 with the COSMIC database, as the input data used to create the COSMIC signatures differs from the real data being analyzed. Therefore, a lower cosine similarity threshold, such as 0.8, is more appropriate for approximating the comparison to the real signatures.

8 CONCLUSION

In this study, we proposed a federated learning system utilizing NMF and AE as mutational signature extraction methods while achieving competitive results compared to centralized learning. We managed to sucessfully extend mutational signature extractions methods to a federated learning setting and compare the performance of these to ones in a centralized learning setting. Our results show that the federated learning setting has a minimal loss in performance compared to the centralized learning setting but at an increased computation time. In addition, the mutational signatures are extracted while preserving privacy of the original data residing on each client. Our approach may be a practical solution of training high quality models on data guarded by privacy restrictions.

However, our research also highlights the need for further research to address the efficiency challenges associated with federated learning on large-scale genomic datasets. Future studies could focus on refining federated learning algorithms to optimize computation time and resource utilization without compromising model performance.

In conclusion, our findings demonstrate the potential of federated learning as a privacy-preserving approach for collaborative model training across decentralized data sources. By extending mutational signature extraction methods to a federated learning framework, we can facilitate improved collaboration among legal entities such as healthcare institutions, all while upholding the privacy of patient data.

9 FUTURE DIRECTIONS

This section outlines the directions for further investigation in the domain of mutational signature extraction in a federated learning setting.

9.1 Local Hyperparameter Tuning

By focusing on hyperparameter tuning in a federated learning setting, researchers can explore techniques to optimize the local hyperparameters of clients. Currently, we set the hyperparameters uniformly across all clients, but there is potential to enhance model performance by allowing each client to adapt its hyperparameters to its own local data characteristics and computational resources.

One approach for future researchers could be to develop optimizing algorithms that adjust hyperparameters for each client based on its local dataset and computing capabilities. These algorithms should consider factors such as the distribution of the data, the amount of available computation, and the specific learning dynamics of each client's dataset. By optimizing the hyperparameters to individual clients, the overall performance of the federated model could be improved.

9.2 Handling Aggregation of Local Models Trained on Different Sized Local Datasets

In the current version of our framework, we were unable to train and extract mutational signatures in a federated learning setting where the local datasets are of different dimensions. One potential direction is to develop adaptive aggregation techniques that can accommodate variations in the local dataset sizes. This would enable fair comparions between different federated learning approaches and facilitate the identification of effective strategies for handling dataset dimension discrepancies.

ACKNOWLEDGEMENTS

We would like to thank our supervisors Daniele Dell'Aglio and Rasmus Froberg Brøndum for dedicated supervision and guidance throughout this project.

CODE AVAILABILITY

The source code can be found at https://github.com/dunkedolmer/FedMutSigExtract

REFERENCES

[1] [n.d.]. *Catalogue of Somatic Mutations in Cancer (COSMIC): Single Base Substitution Signatures (SBS) - SBS17b*. https://cancer.sanger.ac.uk/signatures/sbs/sbs17b/

[2] [n.d.]. *Catalogue of Somatic Mutations in Cancer (COSMIC): Single Base Substitution Signatures (SBS) - SBS34*. https://cancer.sanger.ac.uk/signatures/sbs/sbs34/

[3] [n.d.]. Flower: A Friendly Federated Learning Framework. https://flower.ai/.

[4] [n.d.]. Optuna: A Next-generation Hyperparameter Optimization Framework. https://optuna.org/.

[5] 2023. *COSMIC mutational signatures*. https://cancer.sanger.ac.uk/signatures/ Version 3.4, released in October 2023.

[6] 2023. *COSMIC mutational signatures for single base substitutions*. https://cancer.sanger.ac.uk/signatures/sbs/ Version 3.4, released in October 2023.

[7] 2023. International Cancer Genome Consortium (ICGC). Global collaborative initiative for cancer genomics research. https://dcc.icgc.org/.

[8] International Cancer Genome Consortium (ICGC) 2023. *Pan-Cancer Analysis of Whole Genomes (PCAWG)*. International Cancer Genome Consortium (ICGC). https://dcc.icgc.org/pcawg

[9] 2024. *linear sum assignment algorithm*. https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.linear_sum_assignment.html

[10] Ludmil B Alexandrov, Jaegil Kim, Nicholas J Haradhvala, Min Ni Huang, Alvin W T Ng, Arnoud Boot, and et al. 2020. The repertoire of mutational signatures in human cancer. *Nature* (2020).

[11] Ludmil B. Alexandrov, Serena Nik-Zainal, David C. Wedge, Peter J. Campbell, and Michael R. Stratton. 2013. Deciphering Signatures of Mutational Processes Operative in Human Cancer. *Cell Reports* 3, 1 (31 Jan 2013), 246–259. https://doi.org/10.1016/j.celrep.2012.12.008

[12] Ethem Alpaydin. 2020. *Introduction to Machine Learning*. MIT Press, Cambridge.

[13] A Asuncion and D Newman. 2007. Adult Data Set. UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/datasets/Adult

[14] Dor Bank, Noam Koenigstein, and Raja Giryes. 2021. Autoencoders. *arXiv preprint arXiv:2003.05991v2* (2021). https://arxiv.org/abs/2003.05991v2

[15] Dohyun Cha, Sang Hyuk Shin, Sung Huhn Kim, Ji Yun Choi, and In Seok Moon. 2020. Machine learning approach for prediction of hearing preservation in vestibular schwannoma surgery. *Scientific reports* 10, 1 (Apr 2020), 7136. https://doi.org/10.1038/s41598-020-64175-1 [Medline: 32346085].

[16] Dongchul Cha, MinDong Sung, and Yu-Rang Park. 2023. Implementing Vertical Federated Learning Using Autoencoders: Practical Application, Generalizability, and Utility Study. *Journal Name* (2023).

[17] Narjice Chafai, Luigi Bonizzi, Sara Botti, and Bouabid Badaoui. 2023. Emerging applications of machine learning in genomic medicine and healthcare. *Journal of Experimental & Clinical Genomics* (2023), 140–163. https://doi.org/10.1080/10408363.2023.2259466

[18] COSMIC. 2024. SigProfilerAssignment. https://cancer.sanger.ac.uk/signatures/assignment/

[19] Flower. 2024. FedAvg. https://flower.ai/docs/framework/ref-api/flwr.server.strategy.FedAvg.html. Implementation based on https://arxiv.org/abs/1602.05629.

[20] Nicolas Gillis. 2014. The Why and How of Nonnegative Matrix Factorization. *arXiv preprint arXiv:1401.5226* (2014). https://doi.org/10.48550/arXiv.1401.5226

[21] Luciano Grippo and Marco Sciandrone. 2000. On the convergence of the block nonlinear Gauss-Seidel method under convex constraints. *Operations Research Letters* 26, 3 (2000), 127–136.

[22] Andrew Hard, Hadi Rao, Alexander Mathews, and Shilpi Ramaswamy. 2018. Federated Learning for Mobile Keyboard Prediction. *arXiv preprint arXiv:1811.03604* (2018).

[23] Daniel D Lee and H Sebastian Seung. 2000. Algorithms for nonnegative matrix factorization.

[24] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. *arXiv preprint arXiv:1602.05629* (2017).

[25] Guangxiu Pei, Rongjing Hu, Yiwei Dai, Zongpei Zhao, Peng Jia, Yang He, Cheng Hu, Hongwen Kang, and Zhonghu Zhang. 2020. Decoding whole-genome mutational signatures in 37 human pan-cancers by denoising sparse autoencoder neural network. *Oncogene* 39, 26 (2020), 5031–5041. https://doi.org/10.1038/s41388-020-1343-z

[26] Tom J Pollard, Alistair EW Johnson, Jesse D Raffa, Leo Anthony Celi, Roger G Mark, and Omar Badawi. 2018. The eICU Collaborative Research Database, a freely available multi-center database for critical care research. *Scientific data* 5 (Sep 2018), 180178. https://doi.org/10.1038/sdata.2018.178 [Medline: 30204154].

[27] Yuqiu Qian, Conghui Tan, Danhao Ding, Hui Li, and Nikos Mamoulis. 2020. Fast and Secure Distributed Nonnegative Matrix Factorization. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* (2020). https://doi.org/10.48550/arXiv.2009.02845 arXiv:2009.02845 [cs.LG]

[28] Frederik Rasmussen, Casper Gislum, Kevin Risgaard Sinding, Magni Jógvansson Hansen, Mathias Vestergaard Jensen, and Nikolai Eriksen Kure. 2024. SEEF: A Signature Extraction and Evaluation Framework. (2024).

[29] Muhammad Habib ur Rehman and Mohamed Medhat Gaber (Eds.). 2022. *Federated Learning Systems: Towards Next-Generation AI*. Studies in Computational Intelligence, Vol. 965. Springer, Warsaw, Poland.

[30] scikit learn. 2024. Kmeans. https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

[31] Virginia Smith, David Chiang, Maziar Sanjabi, and Ameet Talwalkar. 2017. Federated Multi-Task Learning. *arXiv preprint arXiv:1705.10467* (2017).

[32] The Cancer Genome Atlas Program (TCGA). 2006. The Cancer Genome Atlas (TCGA): A landmark cancer genomics program. NCI and National Human Genome Research Institute joint effort.

[33] Arne Van Hoeck, Nils H Tjoonk, Ruben van Boxtel, and Edwin Cuppen. 2019. Portrait of a cancer: mutational signature analyses for cancer diagnostics. *BMC Cancer* 19, 1 (15 May 2019), 457. https://doi.org/10.1186/s12885-019-5677-2

[34] Wensheng Xia, Ying Li, Lan Zhang, Zhonghai Wu, and Xiaoyong Yuan. 2021. A Vertical Federated Learning Framework for Horizontally Partitioned Labels. *arXiv preprint arXiv:2106.10056* (2021). https://doi.org/10.48550/arXiv.2106.10056

[35] Guoqing Yang, Shuang Li, Hongming Shan, Wenzhe Guo, Haiping Chen, Dongdong Lin, and Yizhou Wang. 2020. Federated Learning in Mobile Edge Networks: A Comprehensive Survey. *IEEE Communications Surveys Tutorials* 22, 3 (2020), 2031–2063. https://doi.org/10.1109/COMST.2020.2962477

[36] Ph.D. Yu, Justine E. 2021. *Beyond the Signature: Exposing Mutational Patterns of Cancer*. https://dceg.cancer.gov/news-events/news/2021/mutational-signatures