future internet

MDPI

Article

HealthFetch: An Influence-Based, Context-Aware Prefetch Scheme in Citizen-Centered Health Storage Clouds

Chrysostomos Symvoulidis [1,2,*], George Marinos [1], Athanasios Kiourtis [1], Argyro Mavrogiorgou [1] and Dimosthenis Kyriazis [1]

[1] Department of Digital Systems, University of Piraeus, 18534 Piraeus, Greece; gmar@unipi.gr (G.M.); kiourtis@unipi.gr (A.K.); margy@unipi.gr (A.M.); dimos@unipi.gr (D.K.)
[2] BYTE Computer S.A., 11741 Athens, Greece; csymvoulidis@byte.gr
* Correspondence: simvoul@unipi.gr

Abstract: Over the past few years, increasing attention has been given to the health sector and the integration of new technologies into it. Cloud computing and storage clouds have become essentially state of the art solutions for other major areas and have started to rapidly make their presence powerful in the health sector as well. More and more companies are working toward a future that will allow healthcare professionals to engage more with such infrastructures, enabling them a vast number of possibilities. While this is a very important step, less attention has been given to the citizens. For this reason, in this paper, a citizen-centered storage cloud solution is proposed that will allow citizens to hold their health data in their own hands while also enabling the exchange of these data with healthcare professionals during emergency situations. Not only that, in order to reduce the health data transmission delay, a novel context-aware prefetch engine enriched with deep learning capabilities is proposed. The proposed prefetch scheme, along with the proposed storage cloud, is put under a two-fold evaluation in several deployment and usage scenarios in order to examine its performance with respect to the data transmission times, while also evaluating its outcomes compared to other state of the art solutions. The results show that the proposed solution shows significant improvement of the download speed when compared with the storage cloud, especially when large data are exchanged. In addition, the results of the proposed scheme evaluation depict that the proposed scheme improves the overall predictions, considering the coefficient of determination ($R^2 > 0.94$) and the mean of errors (RMSE < 1), while also reducing the training data by 12%.

Keywords: data prefetching; data replication; cloud computing; electronic health records; citizen-centered health

check for updates

Citation: Symvoulidis, C.; Marinos, G.; Kiourtis, A.; Mavrogiorgou, A.; Kyriazis, D. HealthFetch: An Influence-Based, Context-Aware Prefetch Scheme in Citizen-Centered Health Storage Clouds. *Future Internet* **2022**, *14*, 112. https://doi.org/10.3390/fi14040112

Academic Editors: Paolo Bellavista and Carlos Filipe Da Silva Portela

Received: 19 February 2022
Accepted: 29 March 2022
Published: 1 April 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

It is undeniable that cloud computing has a prominent role in the way markets and businesses operate. From the banking and finance industry [1] to manufacturing [2], among others, its adoption is helping leading companies improve their performance across a broad range of services. Typically, a new term is also introduced—Industry 4.0 [3]—in order to describe the Fourth Industrial Revolution and to frame the impact of cloud computing and other emerging technologies, such as the Internet of Things (IoT).

This, however, could not leave the health sector unaffected. Over the last few years and the evolution of the healthcare sector along with already established technologies such as cloud and edge computing and the Internet of Things (IoT), a significant turn toward the citizens' health data ownership was identified [4,5]. For this reason, more and more technology companies have been turning toward that direction [6,7] in order to allow their users to own and be responsible for their own Personal Health Records (PHR), and even countries [8] have already developed patient-oriented systems that provide new capabilities to the citizens, namely allowing them to combine health data from various

sources and healthcare institutions in order to provide them complete pictures of their health. However, while the abovementioned initiatives certainly are steps in the right direction, there is still a significant lack of cloud solutions that are citizen- [9] and not healthcare professional-centered [10,11].

Not only that, even the works that put the citizen in the front are yet to be engaged in the health sector in a way that the health data obtained and kept by the citizen can be utilized by healthcare professionals and vice versa. While most of the performed research only tried to either assist solely the healthcare professionals or help citizens and individuals to be engaged with their health data [6,7], our research focuses on merging these two currently separated areas and providing to citizens a better and more secure way of managing their own health data, while also enabling and promoting communication with the professionals.

In addition, to the best of our knowledge, a lack of services that will assist and accelerate overall communication is identified. Although there exist numerous data prefetching and replication methodologies proposed in the state of the art, there exist none that are entirely health data-related.

For this reason, this paper proposes a storage cloud solution allowing the citizens to safely store their health data in an encrypted manner using only their mobile devices. In addition, this research work allows the citizens to share this health data with authorized healthcare professionals (HCPs) via their healthcare system during emergency situations. Not only that, in order to significantly reduce the data transmission delay between the storage cloud and the HCPs' system, we propose a novel prefetch engine scheme integrated within the proposed storage cloud. This prefetch engine scheme interacts with the storage cloud when requests are performed by physicians and uses this sequence of requests in order to predict the upcoming request. Once predicted, the predicted health record is replicated to the deployed prefetch component, while the storage cloud is notified to request the file from it.

The remainder of this paper is constructed as follows. Section 2 presents in detail the state-of-the-art analysis that was conducted prior to the design of both the proposed prefetching scheme as well as the design of the health storage cloud. Section 3 describes the proposed prefetching scheme, the offered storage cloud solution, and how these services are integrated, while Section 4 presents the experiments that were conducted in order to evaluate the prefetching scheme and the overall storage cloud solution and discusses the results. Finally, Section 5 concludes the paper and proposes the next steps based on the current research.

2. Related Works

This section presents in detail the work that has been achieved in the areas of data prefetching and caching as well as data replication for the optimization of performance. In addition, a dedicated section is devoted to analyzing the state of the art when it comes to health data storage and management on the cloud. Based on this analysis, the proposed prefetching scheme is designed.

2.1. Data Prefetching, Caching, and Data Replication

Data prefetching or caching are techniques that are highly used mostly in databases [12,13] in order to boost the execution performance by fetching data in the memory before it is needed. These methodologies are widely used in database systems or the prefetching of small-sized data such as websites [14,15] or Internet of Things (IoT) data [16].

More precisely, in [14], an implementation of caching and prefetching algorithms in a clustered network is proposed that will improve the retrieval of web objects. In a similar manner, Hussien et al. in [15] proposed a prefetching scheme in mobile cloud computing infrastructures enriched with machine learning capabilities. A set of machine learning algorithms are used, such as C4.5 (and its implementation in WEKA, the algorithm J48), naïve Bayes, and random trees that generate decision trees which cluster the users' objects

and predict which one will be requested. Prefetching and caching methodologies are also used in data derived from IoT devices, as also presented in [16]. In more detail, the authors proposed a graph prefetching framework named Grap which analyzes a building's topologies and prefetches location-related IoT data for the user without knowing the user's destination during navigation.

While prefetching works great with files that are smaller in size, when it comes to large files that cannot be manipulated or stored even shortly in the cache, other technologies such as data replication [17] are used instead. Several works exist that tried to utilize this paradigm and adopt it in the cloud and web services domains. To begin with, the authors in [18] proposed a data replication strategy in order to identify the correlation of the data files stored in the cloud using historical data, predicting which data were going to be requested and replicating it in order to reduce transmission delays. In more detail, the proposed data replication strategy in [18] identifies groups of popular files within the cloud infrastructure, namely files with strong correlations among them, and replicates them in the requested site, thus improving both the access time and the bandwidth usage overall.

In a similar manner, the authors in [19] proposed another correlation-aware replication strategy based on a rule-based management system identifying the data to be replicated. They identified high-valued files (i.e., files that were more likely to be requested), and based on this, they identified which files should be replicated in a prefetch pool. In addition, if the storage of the prefetch pool is not enough, a decision is made based on the abovementioned values in order to remove files already existing in the prefetch pool to free space for the new replicas.

2.2. Cloud Solutions for Health Data Storage

This section presents the work that has been achieved in the areas of health-related storage clouds and health data management systems in the cloud. Over the years, several storage cloud solutions were proposed in order to solve various issues related to secure health data storage and health data management in public, private, or hybrid clouds. To begin with, the authors in [20] proposed an approach that manages patients' personal health data in the cloud, while in emergency situations, healthcare professionals with valid identifications can access this data. In addition, major service providers such as Google [6] and Microsoft [7] provided Google Health and Microsoft Cloud for Healthcare, respectively, to their users. These platforms can be used to manage PHR.

When it comes to Electronic Health Records (EHR), meaning health data that are managed by health professionals, there is an even wider area of conducted research. More specifically, a significant amount of research in this area already started over a decade ago [21–23], yet several problems need to be solved. Starting with [24], the authors proposed a distributed storage model for EHR using Apache HBase [25], a column-oriented database built on top of a Hadoop Distributed File System [26]. In addition, there exists the research in [27], where an EHR storage cloud system was proposed with the main goal of tackling issues related to privacy and security when EHR sharing among different entities occurs with the use of a blockchain.

Additionally, another research work by Song et al. [28] proposed a cloud-based PHR system architecture that allows sharing of interoperable health data with clinicians. The proposed system can be used to upload files of any type (files, medical images, video files, etc.) in such a way that it can provide easy access to storage for both citizens and clinicians.

2.3. Advancements beyond the Related Work

Based on the above, it can be understood that a lot of effort has been made in the scientific community in order to provide healthcare professionals, healthcare institutions, and citizens abilities related to secure health data storage and sharing, although significantly less attention has been given to communication between the professionals and the individuals. For this reason, this paper proposes a novel data prefetching scheme that can be adopted by health data storage cloud solutions, which will facilitate the accelerated

exchange of health data (both EHR and PHR) among citizens and healthcare professionals. It is our intention to exploit this prefetching scheme in emergency cases, since it is in such events where such a methodology is utilized and needed the most.

In more detail, in this work, both a prefetching scheme to enable faster transmission times for EHR exchange adopted by a storage cloud solution as well as a cloud storage solution, which can be used by citizens in order to securely store their health data and by healthcare professionals that can access this information when needed, are examined.

3. Materials and Methods

This paper proposes a novel prefetching scheme for distributed file systems in cloud computing environments and storage clouds in order to eventuate better I/O performance and reduce the retrieval and overall transmission times of the requested files.

In this section, we first introduce the proposed prefetching scheme, along with the related prediction algorithms. Consequently, the architecture of the storage cloud solution that the prefetching scheme is integrated with is presented. (A graphical depiction of the high-level architecture of the proposed storage cloud is illustrated in Figure 1.)

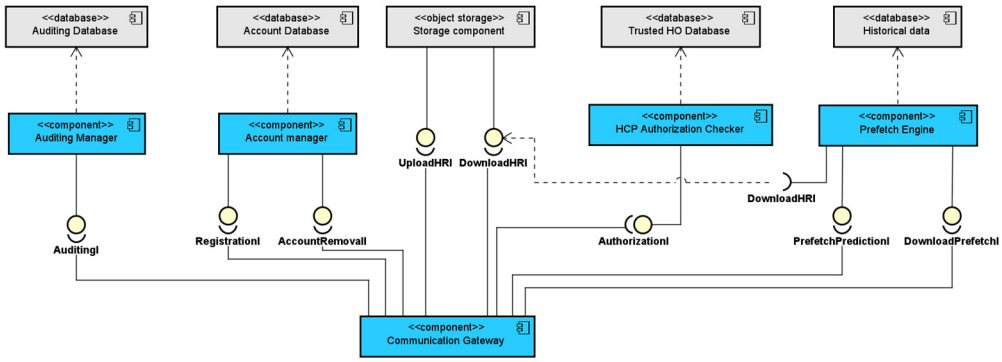

Figure 1. High-level view of the proposed cloud storage solution integrated with the novel prefetch engine mechanism.

3.1. The Context-Aware Prefetch Scheme

The proposed prefetch scheme aims at providing recommendations regarding the files that should be cached, taking into account the context behind each individual request. In this case, the term "context" refers to the information included in the requests of the users that can be used in such a way to facilitate the prefetch scheme in decision making and make customized predictions for each user. In other words, the proposed prefetch scheme is able to extract information from the incoming requests that will allow it to provide better predictions. The workflow can be split into two periods: (1) the training phase, which is performed offline (i.e., the *offline phase)*, and (2) the recommendation and pre-fetching phase triggered every time a request is received, which is performed online (i.e., the *online phase*). A visual representation of the overall workflow of the prefetch scheme is depicted in Figure 2.

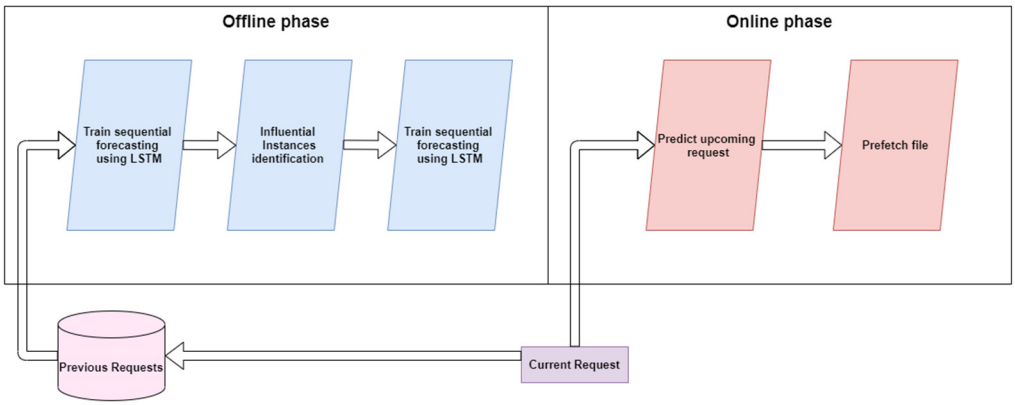

Figure 2. The prefetch scheme's overall workflow, including influential instances.

3.1.1. The Offline Phase

The offline phase begins with the training of a prediction model using a data set related to requests performed at an earlier time. In more detail, the incoming requests are considered elements of a sequence, where the first element of the sequence is the first request coming from a specific user and the last element is the latest received request coming from the same user. In parallel with the process of deep learning model training, which will be further analyzed in the following paragraphs, we continuously monitored the performed requests that were performed by a user. By monitoring this process, we finally came up with a sequence of requested files along with the timestamp of each request. Thus, we predicted the upcoming request based on the previous data.

The proposed storage cloud architecture in order to maintain data privacy does not allow the cloud provider to have access to the actual data, other than some metadata related to the files including the name, size, the bucket they are stored in, and other auditing information such as the changes to a file, the user who uploaded or downloaded a file, and the latest version of the file. For this reason, the prefetch recommendation is performed solely on the requests of the user, and this additional metadata and auditing information can contextualize the prediction. We rely on a sequential modeling predictive ability in order to capture the temporal relationships between the sequence of the requests performed by a user in order to predict which file will be requested afterward and, as a result, be able to achieve better performance with regard to transmission times.

The preferred model involves long short-term memory (LSTM) [29], as opposed to other statistical models such as ARMA [30], ARIMA [31], or other more robust approaches such as a recurrent neural network (RNN) [32], since the LSTM model has been proven to outperform them. In more detail, there were two reasons behind the use of the LSTM model. First of all, in our experiments, LSTM outperformed classical statistical methods like ARIMA, while this architecture is known to be able to better capture more long-term relationships compared with basic RNN models based on their internal architectures [33]. In addition, the use of LSTM manages to solve the problem of vanishing gradient descent as opposed to an RNN architecture [34].

As far as its architecture is concerned, this model is a seven-layer model consisting of six LSTM layers and a final dense layer, as depicted in Figure 3. In order to avoid overfitting, dropout layers were introduced after each LSTM layer with a rate of 0.75 for each one except the last layer, which had a rate of 0.5. The selection of those rates was performed after experimentations with the testing data sets.

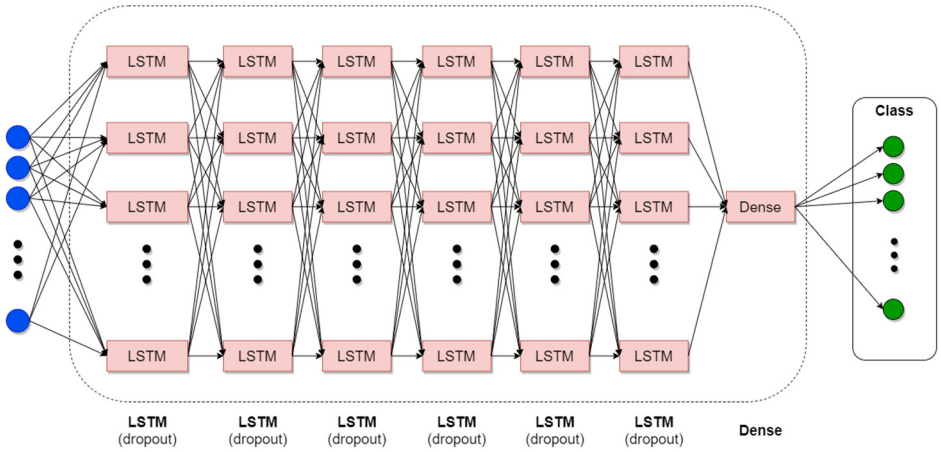

Figure 3. LSTM for time series classification model architecture.

Once the model is trained, the identification of the *influential instances* of the model step is instantiated. These are the instances (i.e., the requests performed by the user) that have a strong effect on the trained model, and while commonly used for machine learning interpretability purposes, the use of this methodology within this scheme is to utilize it in order to provide context-aware predictions and improve the model [29], an idea that is also used in the area of visual analytics on electronic medical records when using an RNN [35].

The core idea behind the influential instances is that an instance is considered influential if once it is deducted from the training set, and the model is re-trained, a significant change is established in the model, and the model's parameters (i.e., model weights) are different while the predictions of the model also change. The greater the identified change to the model and its parameters, the more influential the instance. Hence, in order to examine whether an instance is influential or not, the model needs to be trained with and without this specific instance, and the results should be compared. For instance, in the current scenario, the LSTM model needs to be trained with and without an instance in order to examine whether this instance can be considered influential or not.

As mentioned above, the influential instances are usually used in order to explain the outcomes of machine learning or deep learning models, which are difficult to interpret. The current solution, though, differentiates from the current way the influential instances are used, since it utilizes the influential instances in a way that will empower the LSTM model through the provision of additional information with respect to the data. In more detail, it will provide the model a list of instances that may be more significant than others, which the model may use in order to improve itself. The way these influential instances are identified is shown below.

The effectiveness of an instance i on the model's parameters can be measured using $DFBETA$, which is defined as follows [36]:

$$DFBETA_i = \beta - \beta^{(-i)} \tag{1}$$

where β is the weight vector of the initial model that is trained with all instances and $\beta^{(-i)}$ is the weight vector when the model is trained without the ith instance. However, the calculation of $DFBETA$ alone cannot provide contextual information regarding the influence of a specific instance on the model and its predictions. Additional metrics need to be used. For this reason, the root mean square error (RMSE) is also calculated as shown below:

$$RMSE_i = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(y_j - \hat{y}_j\right)^2} \tag{2}$$

where i is the instance that is removed in order to calculate its influence, n is the number of instances, y_j is the correct target value, and $\hat{y}_j$ is the prediction of the model.

With the use of *DFBETA* and the calculation of the *RMSE* for each instance, the influence of the instance on the model can be evaluated. With this information, the instances are sorted based on significance, and the most influential ones are then kept for the next step.

As depicted in Figure 4, once *DFBETA* and the *RMSE* are calculated, they are then provided as input to the next step, which regards the creation of clusters of important instances with respect to the ones that were previously identified. This is performed with the use of the k-means [37] algorithm. In more detail, these instances are provided to the k-means algorithm as initialization points for the centroids, and based on them, the training of the algorithm is instantiated. One important note is that by default, the k-means algorithm utilizes the Euclidean distance for cluster creation, but since the current problem relates to time series data, the Euclidean distance is not useful. For this reason, the dynamic time warping matching [38] distance metric is utilized. This metric measures the similarity between sequences that may vary in steps and, in essence, can calculate an optimal match between them.

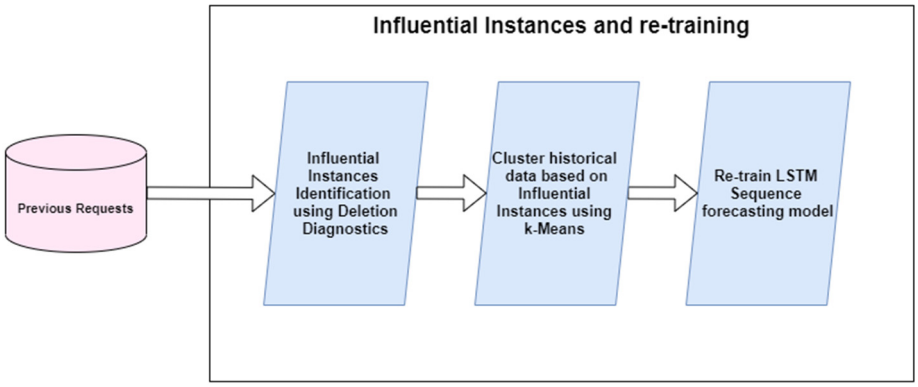

Figure 4. Influential instances identification workflow.

Consequently, the remaining instances that were not initially categorized as influential are forwarded to the k-means algorithm, and if the distance from one of the centroids is smaller than a given number, it is considered part of this; otherwise, it is considered an outlier, as also depicted in Algorithm 1.

The above step is repeated for all remaining instances in the initial data set of sequences. Once this is over, the instances that belong to either of the clusters are used as input for the LSTM model for re-training.

The offline phase is repeated daily, while all incoming requests are stored in a dedicated database in order to re-apply the above shown scheme with the new data. This database includes historical data of the requests that are made by dedicated healthcare professionals to the cloud. In detail, what is stored is a timestamp of the request, the requested file, and the session in order to group all requests made during the same emergency.

3.1.2. The Online Phase

The online phase regards the step that is triggered once a sequence of requests is sent to the storage cloud by a single user. In more detail, based on this sequence, the LSTM model predicts the upcoming request and requests the storage cloud to cache this data for transmission.

Once the request is received and the requested data match the prefetched data, the actual exchange of them is instantiated directly from the prefetch component. On the other hand, if the request made by the user does not match the prefetched data, the prefetched data are dropped in order to save space, and the data are requested from the Storage Component instead. In both cases, the sequences of the requests are stored in a dedicated database that keeps all historical data with respect to the sequences of requests, as seen in Figure 2.

Algorithm 1 Prefetch scheme workflow

Input: HTTP requests data set from HCPs in the form of time series (D)
Auxiliary Variables: $GIVEN\_RMSE$, $GIVEN\_DISTANCE$
Output: Predicted health record to be prefetched (y)
Begin *HealthFetch*

1. LSTM(D)
2. **for each *i* in D_i : $i \notin D$:**
3. LSTM(D_i)
4. $DFBETA_i = \beta - \beta^{(-i)}$
5. $RMSE_i = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(y_j - \hat{y}_j\right)^2}$
6. **end for**
7. **for *i* in *D*:**
8. **if (*DFBETA*$_i$>*AVG*(*distance*, *D*) AND *RMSE*$_i$>*GIVEN_RMSE*):**
9. $i \in InfluentialInstances$
10. **end if**
11. **end for**
12. **k-Means(**$InfluentialInstances$, DTW**)**
13. **for each *j* in D_j : $j \notin$ *InfluentialInstances*:**
14. **for each *i* in *InfluentialInstances*:**
15. **if *distance*(*j*, *i*)<*GIVEN_DISTANCE*:**
16. $j \in InfluentialInstances$
17. **end if**
18. **end for**
19. **end for**
20. $y =$LSTM($InfluentialInstances$)

End *HealthFetch*

3.2. The Health Storage Cloud

The proposed prefetching scheme was applied to a distributed storage cloud solution used for the backup of health data by citizens while allowing access to healthcare professionals during emergency situations [39,40]. The storage cloud, as also seen in Figure 1, was composed of the following components:

- The *Account Management Component*, which was used for the management of the citizens' accounts and account policies, along with the management of the healthcare institutions' temporary account creation during emergency situations;
- The *Storage Component*, which was used for the storage of the citizens' health data;
- The *HCP Authorization Component*, which was used for the authorization of healthcare institutions during emergency situations;
- The *Auditing Manager Component*, which was used for keeping logs of all actions performed by any user of the storage cloud (i.e., both citizens and healthcare professionals from authorized institutions);
- The *Communication Gateway Component*, which handled all incoming requests from the client side (i.e., both citizens and healthcare professionals);
- The *Prefetch Engine Component*, which regarded the implementation of the proposed prefetch scheme that predicted the upcoming requests from the users and prefetched the corresponding health data.

3.2.1. The Account Management Component

This component is responsible for managing the accounts of the users of the storage cloud for both citizens and healthcare professionals. Starting with the citizens, once they request to use the proposed storage cloud service, an account is created that is linked to a bucket on the Storage Component. A bucket is essentially a directory, visible only through this user's account, which the user utilizes to store their encrypted health data. This bucket is also visible for temporary accounts of authorized healthcare professionals

during emergency situations. An important note is that both read and write permissions are given only to the citizen's account, while all other accounts (i.e., temporary accounts of healthcare professionals) have only read permissions. These permissions are given to the citizen's account using a dedicated policy, such as the one presented in Figure 5.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "*"
        ]
      },
      "Action": [
        "s3:GetBucketLocation",
        "s3:ListBucket",
        "s3:ListBucketMultipartUploads"
        "s3:AbortMultipartUpload",
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:ListMultipartUploadParts",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::${aws:citizen}/*",
        "arn:aws:s3:::${aws:citizen}*"
      ]
    },
    {
       // Additional policies
    }
}
```

Figure 5. Sample policy for citizen's account on the cloud.

On the other hand, there may exist temporary accounts created during a health-related emergency. Those accounts can be utilized by healthcare professionals from authorized institutions (more information on how the healthcare professionals are authorized is presented in Section 3.2.3). In addition to this account, an emergency bucket is also created that is linked to both the temporary account of the healthcare institution and the citizen's main account. This bucket is used by the healthcare professionals to upload (encrypted) health data related to the emergency or the discharge report once the emergency is over and the citizen is discharged. In this bucket, both the temporary account of the healthcare institution and the citizen's account have read and write permissions.

3.2.2. The Storage Component

This component is responsible for storing the health data of a citizen. Its purpose is to ensure that the health data are not only stored safely but also available instantaneously, especially during an emergency situation where every moment is crucial, and thus any potential delays should be subsided. The Storage Component is based on an object storage solution: MinIO Object Storage [41]. The rationale behind the use of an object store instead of a conventional NoSQL-based solution such as MongoDB/GridFS [42–44] or an SQL-based system is manifold. The health data stored in the storage cloud should be encrypted on the client side (i.e., the citizen) as well as the storage cloud side so that the cloud provider does not have access to it and to ensure that even in cases where an unauthorized entity manages to access the storage cloud, they will be unable to access the health data per se. Hence, the data stored in the cloud are, in essence, encrypted bundles. Given the fact that the health data stored in object storage are stored as objects, this solution suited our needs. In addition, the proposed storage cloud service can also be deployed by individuals as a self-hosted service. Another important feature that the object storage solution we utilized has is the fact that it can also be deployed easily in commodity hardware. Finally, high

availability assurance, along with the ease of scalability of MinIO, played a very important role in using it.

As far as the architecture of this component is concerned, as already mentioned above, specific policies are applied to the various accounts that are created, which are managed by the Account Management Component (see Section 3.2.1).

3.2.3. The HCP Authorization Component

The HCP Authorization Component is responsible for probing whether a healthcare institution is trusted to access the citizen's health data during an emergency. The healthcare professionals curing the citizen perform a request to the storage cloud in order to access the citizen's health data (i.e., a temporary account is to be created). This request should include a set of authorization attributes derived from a trusted third-party certification authority. Consequently, the HCP Authorization Component contacts this trusted authority and forwards those attributes to it. The certification authority then evaluates those attributes and may either approve or decline the request. If the request is accepted, then the HCP Authorization Component allows the creation of the temporary account, and the account is created. If the request is declined, then a response is sent to the Communication Gateway Component (see Section 3.2.5) in order to reply to the requester that their request was rejected.

3.2.4. The Auditing Component

This component keeps logs of all actions performed with respect to the encrypted health data (uploads, downloads, modifications, and uploads of newer versions), the requests to access the citizen's health data (both approved and rejected), and the emergency accounts that were created along with the data that were downloaded or uploaded. In more detail, the information that the auditing manager component keeps includes the following:

- List of uploads of encrypted health data performed by the citizen, including a timestamp;
- List of downloads of encrypted health data performed by the citizen, including a timestamp;
- List of approved requests by healthcare institutions;
- List of healthcare professionals and physicians that were granted access to the storage cloud through the temporary account of the healthcare institution he or she is working for;
- List of rejected requests and who they were from;
- List of downloads of encrypted health data performed by physicians from a trusted healthcare institution, including a timestamp;
- List of uploads of encrypted health data performed by physicians from a trusted healthcare institution, including a timestamp.

The above-mentioned data are kept in a MongoDB [45] database separate from the Storage Component and can always be made available to the citizen. A sample of the logs that the auditing manager creates and stores in the database is shown in Figure 6.

```
"auditing": {
    "hr_info": {
        "uploaded": [{
            "uploaded_hr": "$hrName",
            "hr_uploaded_on": "$timestamp"
        },
        ...
        ],
        "downloaded": [{
            "downloaded_hr": "$hrName",
            "hr_downloaded_on": "$timestamp"
        },
        ...
        ]
    },
    "hco": {
        "granted_access": [{
            "healthcare_organization_name": "$hcoName",
            "granted_access_on": "$timestamp",
            "healthcare_professionals_granted_access": [
                "$hcp1Name",
                "$hcp2Name",
                ...
            ]
        },
        ...
        ],
        "downloaded": [{
            "healthcare_organization_name": "$hcoName",
            "healthcare_professional_name": "$hcpName",
            "downloaded_hr_on": "$timestamp",
            "downloaded_hr": "$hrName"
        },
        ...
        ],
        "uploaded": [{
            "healthcare_organization_name": "$hcoName",
            "healthcare_professional_name": "$hcpName",
            "downloaded_hr_on": "$timestamp",
            "downloaded_hr": "$hrName"
        },
        ...
        ]
    }
}
```

Figure 6. Auditing information kept by the Auditing Component that can be provided to the storage cloud's users.

3.2.5. The Communication Gateway Component

The Communication Gateway component is responsible for managing any incoming requests to the storage cloud. These requests may come from both the citizens and physicians during an emergency. It is a Flask service [46] that responds to HTTP requests, (A sample HTTP request is shown in Figure 7.) and its main functionalities are split into two main categories: (1) the functionalities that can be performed by a citizen, such as a request to create an account, a request to upload an encrypted health record, or a request to download the auditing information collected by the Auditing Component, and (2) the functionalities performed by healthcare professionals on behalf of a healthcare institution. These functionalities may include the initial request to access a citizen's health data, a download request of a specific health record, or a request to upload a discharge report once the emergency is over and the citizen is discharged from the institution.

```
GET /hcp/{bucketName}/{objectName}
HTTP/1.1
Host: [BASE URL]:8080
Authorization: [AUTHORIZATION_TOKEN]
    Authorization: [AUTHORIZATION_TOKEN]
        Authorization: [AUTHORIZATION_TOKEN]
            Authorization: [AUTHORIZATION_TOKEN]
                Authorization: [AUTHORIZATION_TOKEN]
```

Figure 7. Sample of requests received by the storage cloud.

The endpoints that implement these functionalities are also categorized based on the entity that performs the request. All requests coming from citizens using the storage cloud are placed under the /citizen/ path, while all requests coming from the personnel of healthcare institutions are placed under the /hcp/ path.

3.2.6. The Prefetch Engine Component

The Prefetch Engine Component regards the implementation of the component of the prefetch scheme as presented above. This engine is responsible for providing recommendations regarding the data an HCP may request from the storage cloud before the actual request in order to cache it and have it ready for transmission once it is eventually requested.

The request coming from the HCPs during an emergency situation are in the form of an HTTP request (as shown in Figure 7). These requests are stacked in the form of a sequence for each HCP, and once this sequence length exceeds a given number, the prefetch engine is triggered. Then, based on this sequence of requests, the prefetch engine, whose training phase is performed offline, predicts the next health data resource, as also shown in Figure 8, which will be requested by the HCP, and caches it. In the case where the resource to be cached is very large, the resource is replicated on the prefetch engine in order to be forwarded eventually to the Gateway Component. This architectural choice is taken because the Prefetch Engine Component should be deployed by the Gateway Component, and hence the transmission delay between the Prefetch Engine Component and the Gateway Component is significantly lower when compared with the transmission delay between the Storage Component and the Gateway Component.

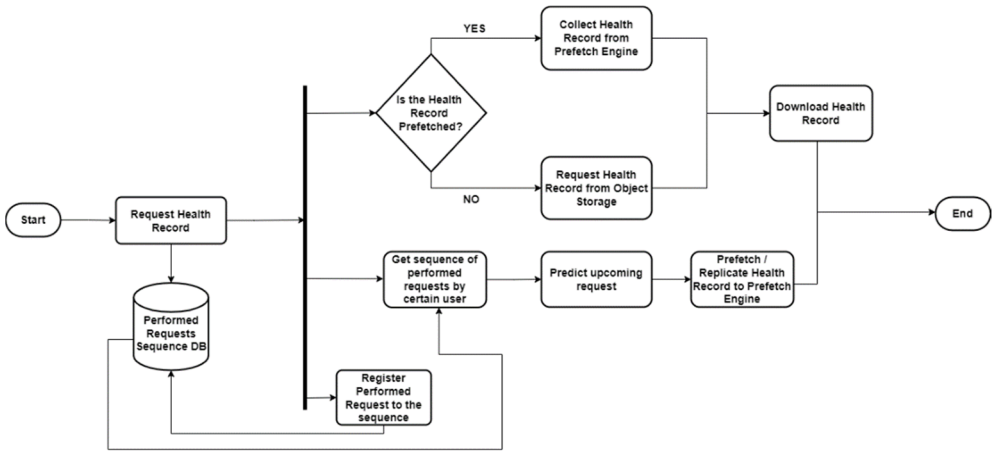

Figure 8. Flowchart depicting the behavior of the prefetch component and the way health data are downloaded when prefetched and when not prefetched.

In both cases, the Gateway Component is notified to request the resource from the prefetch engine instead of the Storage Component. Once the request is finally performed by the user, the data, instead of being transferred from the Storage Component that it was initially stored in, are forwarded from the prefetching component straight to the HCP's application. In addition, in the case where the resource is not requested by the HCP, then it is dropped from the Prefetch Engine Component to save resources.

4. Results

This section describes in detail the experiments that were conducted in order to test the proposed storage cloud and the prefetch engine. The motivation behind the experiments was twofold; first to showcase that with the use of the prefetch engine, the transmission of the data from the storage cloud to the end user was performed significantly faster, and secondly to examine whether the proposed prefetching scheme provided better results when compared with a state-of-the-art solution, such as the use of an LSTM model. In

order to evaluate the proposed prefetch engine, various deployment scenarios were used in order to reflect several use cases that may occur.

4.1. Performance Evaluation

As described above, one of the evaluation tests that the prefetch scheme and the storage cloud was put under had to do with the performance of the proposed prefetch engine and whether it improved the overall delay of the storage cloud.

Starting with the testing regarding the health data transmission, the experiments were conducted in three different deployment configurations. The first configuration had the storage cloud solution deployed in two virtual machines (VMs) in the same infrastructure in order to reflect a typical service deployed in a cloud infrastructure with the following hardware: 2xCPU Intel Xeon CPU E5-2620 v2 @ 2.10 GHz, 8 GB of DDR3 memory, and 60 GB of Hard Disk Drive (HDD). The second experiment was executed with the cloud solution being deployed in one VM with the aforementioned resources in order to reflect deployment in constrained environments such as an edge node, while in the third solution, the storage was deployed in two VMs but in different data centers, again with the similar resources, in order to examine the performance of the prefetch engine in a distributed environment. Regarding the networking abilities of the experimental set-up, for all nodes, the network speed was the same. In particular, the download speed (tested) was 150 Mbps, while the upload speed was 150 Mbps as well. The reason we performed those experiments on different infrastructures was to benchmark the storage cloud in various conditions and draw safe conclusions with respect to the usage of the prefetch engine.

The performed tests evaluated the transmission delay of encrypted health records of various sizes. To be more precise, the download time as shown in the following figures represents the total average time it took for an encrypted health record to be downloaded, starting from the moment the request was performed on the client side (i.e., the HCP side) and ending when the download was complete. In that time, we did not take under consideration the time it took to decrypt the file, since this had to do with the computing power of the client side. In addition, the transmission time was irrelevant to the type of file being exchanged (i.e., either encrypted or decrypted).

In our experiments, we categorized the data to be transmitted by size (5 MB, 50 MB, 500 MB, and 1 GB), and we identified that the majority of the health data stored in the proposed storage cloud could be classified under the presented categories. There exist several types of health data that are significantly small in size (such as a patient resource [47]) and should not exceed the ceiling of 5 MB, while the majority of the exchanged health care data (such as a bundle [48] or a composition [49]) may be around 50 and 500 MB, respectively, depending on the content they include. In addition, the storage cloud also allows the storage of even larger health data types such as DICOM [50] medical studies that may include several projection images and can reach or even surpass 1 GB in size.

The following figures depict the results on the abovementioned configurations.

More specifically, in Figure 9, the average delay (in ms) of health records of various sizes is presented when the proposed storage cloud solution was deployed in multi-cloud environments. The use of a prefetching mechanism significantly reduced the transmission time of a health record.

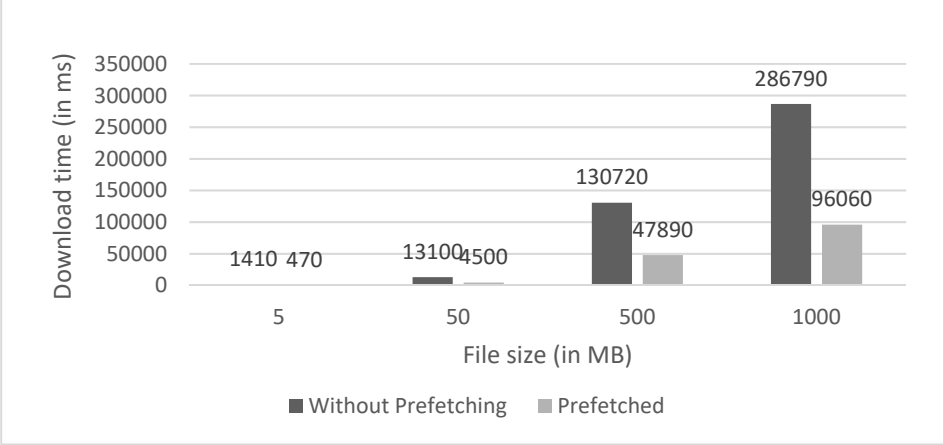

Figure 9. Average download times (in ms) of health record resources with various sizes (5 MB, 50 MB, 500 MB, and 1 GB) when the storage cloud and the prefetch engine were deployed in infrastructures with different physical locations.

Similarly, Figure 10 presents the results in the same tests but with the difference that the storage cloud was deployed in the same cloud infrastructure but in different virtual machines. Again, there was a significant reduction in the transmission time of health data when the data were prefetched.

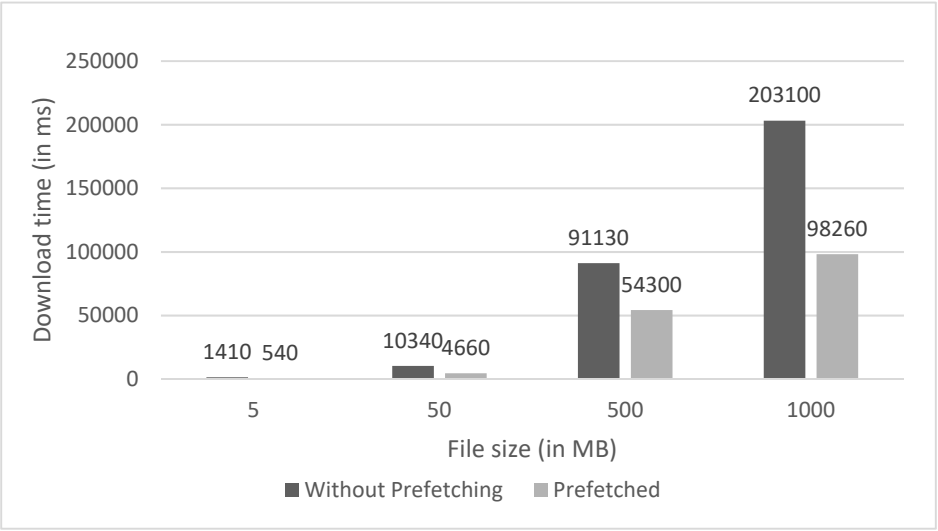

Figure 10. Average download times (in ms) of health record resources with various sizes (5 MB, 50 MB, 500 MB, and 1 GB) when the storage cloud and the prefetch engine were deployed in the same infrastructure but in different nodes (i.e., VMs).

The same results were observed from the results shown in Figure 11, where the transmission of health data in ms is again presented. There, the storage cloud was deployed in one VM. Once more, the results show a significant reduction in the delay time when the data were prefetched when compared with the transmission time of the non-prefetched data. In all figures, the average delay for over 1000 sequential requests/file made to the storage cloud is presented.

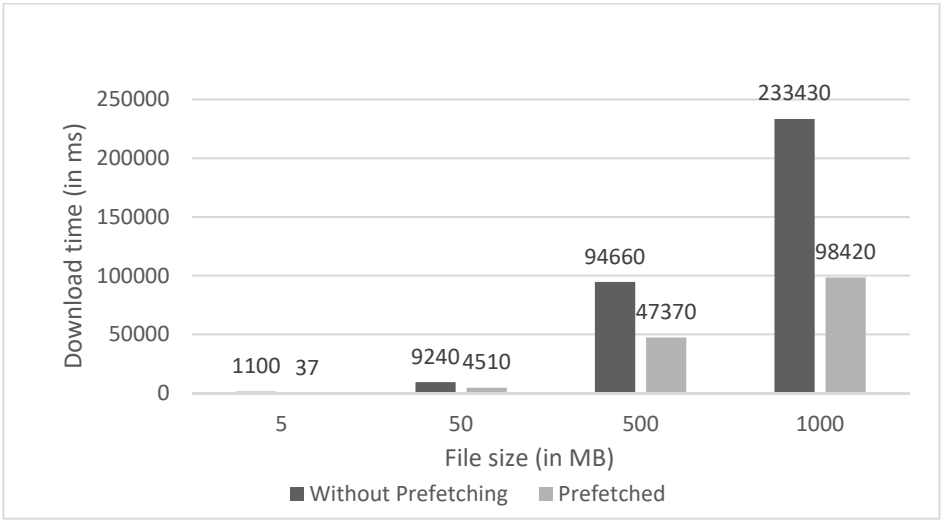

Figure 11. Average download times (in ms) of health record resources with various sizes (5 MB, 50 MB, 500 MB, and 1 GB) when the storage cloud and the prefetch engine were deployed in the same node (i.e., VM).

The abovementioned experiments, as mentioned, concerned sequential requests. For this reason, another set of experiments was also conducted in order to evaluate the performance of the prefetch-enabled storage cloud when parallel requests were made. What is important to note is that the following experiments were conducted in the third deployment configuration (i.e., when the cloud was deployed in different data centers) since, to the best of our knowledge, this is the most common situation when deploying services in a cloud infrastructure. Thus, additional scenarios were deployed, which are presented below. Let us start with the results depicted in Figure 12, where the transmission times for downloading several encrypted health records are shown when performing simultaneous requests. What can be observed is that the delay was significantly increased when the number of requests was bigger, but there was an important reduction in the transmission time when the data were prefetched.

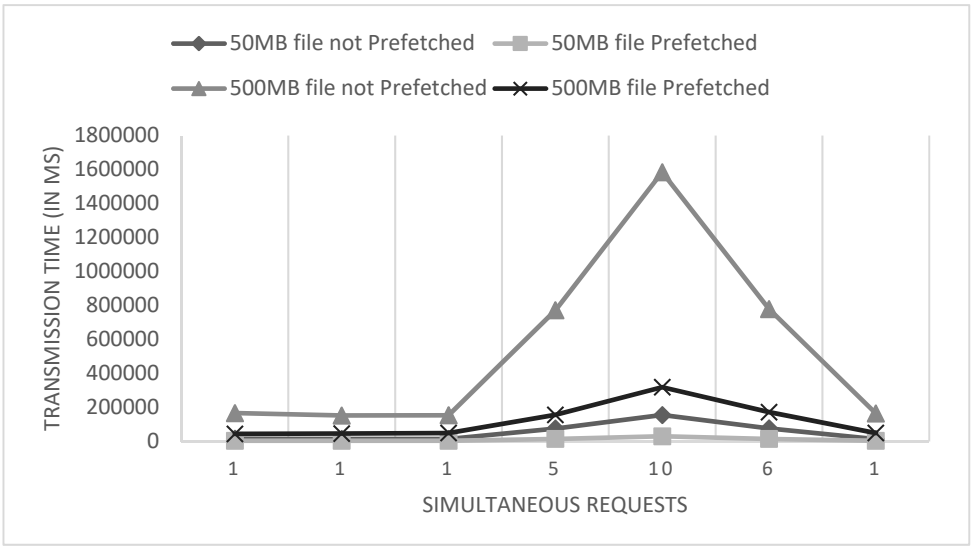

Figure 12. Average download times (in ms) of different encrypted health records while performing simultaneous requests to the deployed storage cloud and deployed in different data centers.

The final experiment with respect to the performance of the prefetch engine had to do with its behavior when performing individual requests asking for different files each time,

and the results are shown in Figure 13 below. Similarly, there was a considerable reduction in the overall transmission time when the prefetching scheme was enabled, leading to the conclusion that the scheme optimized the performance of the storage cloud even in cases where multiple requests were performed.

Figure 13. Average download times of different encrypted health records while performing simultaneous requests asking for different files to the deployed to the storage cloud and deployed in different data centers.

4.2. Model Optimization Using Prefetch Scheme Evaluation

For the evaluation of the proposed prefetch scheme, we utilized a data set consisting of 800 observations that represent a set of requests performed for the cloud service in sequence form. In order to evaluate whether the proposed scheme actually performed better, we tested the performance of the LSTM network with and without the usage of the scheme. In both cases, the data set was split into the same train and test sets, where the train set was comprised of 650 observations and the test set was comprised of 150 observations.

In Figures 14 and 15, the RMSE of the train and test sets with and without the use of the prefetch scheme are presented accordingly. We identified that the prefetch scheme, even though it took more epochs to converge, over the total 25 epochs managed better results against the LSTM without the use of the scheme, with even a significant reduction in the training data set. In both cases, we used early stopping as a form of regularization in order to avoid overfitting, and hence why the training phase stopped at 25 epochs. More details with respect to the performance of the LSTM network with and without the prefetch scheme are presented in Table 1.

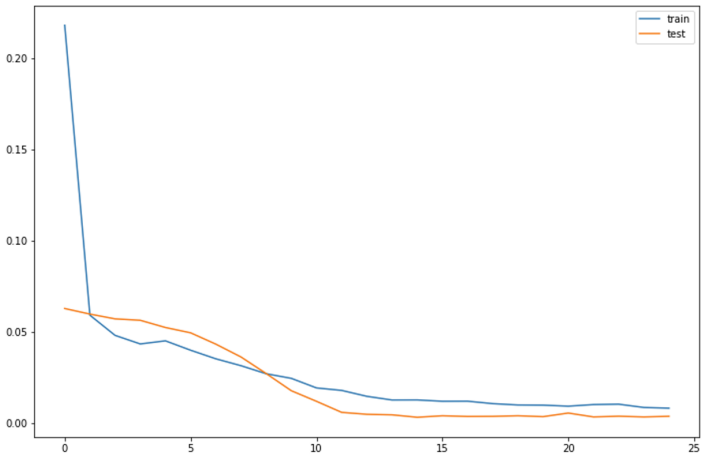

Figure 14. RMSE for train and test sets over 25 epochs without the usage of the proposed prefetch scheme.

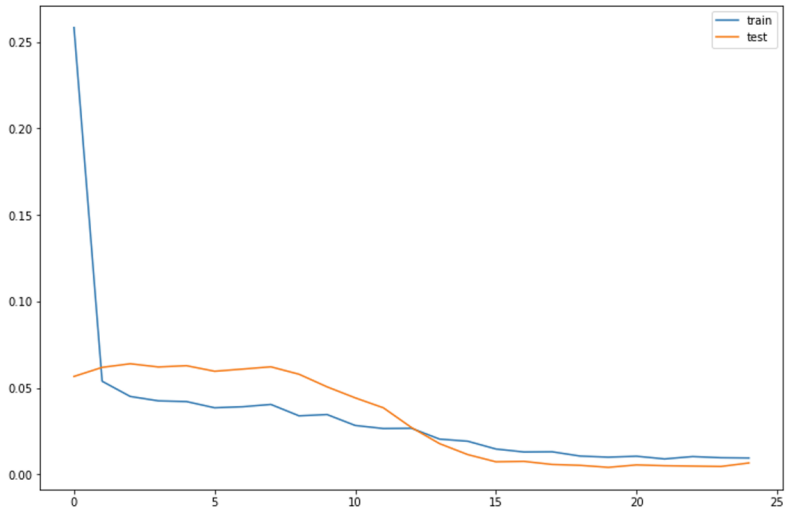

Figure 15. RMSE for train and test sets over 25 epochs with the usage of the proposed prefetch scheme. After usage of the prefetch scheme, the train set was reduced by 20%.

Table 1. RMSE and R^2 of the LSTM network with and without the usage of the proposed prefetch scheme.

Prefetch Scheme Used	# of Observations in Train Set	RMSE	R^2
N	650	0.06	0.954
Y	535	0.048	0.943

4.3. Discussion of Results and Limitations

The above tests tested the proposed prefetch scheme in two main concepts: its performance when utilized and whether the proposed architecture presented any advancements when compared with other more traditional ways, such as a plain LSTM network. Starting with the performance evaluation, several tests were executed, and the results show that when the prefetch scheme was utilized, a high decrease in the average transmission delay was observed, making the overall health data exchange process faster and more agile.

To begin with, as depicted in Figures 9–11, in all deployment configurations, it was identified that the prefetch scheme facilitated the reduction in the transmission times. However, in the last configuration where the proposed storage cloud solution was deployed in different locations, the difference when measuring the download time when a file was prefetched, compared with the download time when the file was not prefetched, was greater than those of the first two configurations. In addition, as also depicted in Figures 12 and 13, when the number of simultaneous requests increased, the transmission time increased significantly, especially when the requested files were not previously prefetched, while in the case where the data were prefetched, the overall data exchange delay was considerably reduced. This can occurred for three reasons. The first reason is related to the prefetching of the data, since when the data were not prefetched prior to the request performed by the user, the overall download time included the time the Storage Component took to collect them from the disk and then transmit them. On the other hand, in the case where the data were already prefetched, this delay was eliminated, since the file was already cached in the Prefetch Engine Component. Moreover, in the current infrastructure, the drives that the proposed service used were HDDs, and thus the retrieval of the files was significantly slower compared with when the data were already cached. In addition, in all configurations, the Prefetch Engine Component was deployed on the same node as the Gateway Component, making communication even faster.

The second evaluation had to do with the proposed scheme and whether it produced better results when comparing it to an LSTM model without utilizing the prefetch scheme. First, as already established in the literature [30–33], the use of an LSTM model already

surpassed the use of other traditional methods, such as the statistical methods of ARMA and ARIMA as well as the use of an RNN model. Yet, with the execution of the evaluation tests shown in Section 4.2, it was our intention to evaluate the proposed prefetch scheme with the current state of the art. In addition, the comparison between the LSTM models with and without the use of the prefetch scheme should highlight the prefetching scheme's importance. Here, some challenges were raised and needed to be solved, with most of them related to the data set. The collected data set was produced after weeks of gathering data from requests. Although the data set size was not big, we managed to utilize it in order to test the proposed scheme, and again, the results show that there was a significant increase in the RMSE and the R^2 metrics even after reduction of the data set was performed. Based on this, we are confident that the proposed scheme would produce even better results if a larger data set could be used, since it was observed that the reduction in the data set size reached around 12%, yet the metrics (RMSE and R^2) also showed an increase in the network's performance.

Regarding the limitations and requirements of the proposed storage cloud architecture and the proposed prefetch scheme, the prefetch scheme requires the user to perform at least five requests in order to predict the upcoming file. Unless these requests are performed, no prediction is made. This choice was made because after evaluating the prefetching scheme, it was discovered that the predictions were not accurate, and thus the process of prefetching a file and caching it would be not useful. Another limitation of the current architecture regards its inability to share the prefetched files with users other than the one for whom it was prefetched. As already described in Section 3.2.2, the files stored in the proposed storage cloud are also encrypted on the client side in order to ensure that the cloud provider cannot have access to the user's health data. Hence, the data were exclusively related to one patient. Thus, in a case where two healthcare professionals perform a request for the same type of health data but for a different patient, even if the data are prefetched for one healthcare professional, the other healthcare professional will not be able to utilize the prefetched files, since they regard a different patient. Another limitation of the proposed solution concerns what happens to data once they are prefetched. As presented in Section 3.2.6, if the data are requested by the user, they are transmitted to them and then dropped, while in the case where the prefetched data are not requested by the user, they are also removed from the Prefetch Engine Component in order to save resources.

Considering the overall issues that occurred throughout this research, we targeted a list of obstacles related with the fact that the proposed storage solution and the integrated prefetching scheme were not yet tested by actual users and healthcare professionals. In fact, the data used for the evaluation were generated based on actual traffic. Moreover, another issue is related to the infrastructure used for the evaluation being self-hosted. It is within our intentions to also assess the proposed solution in cloud infrastructures as well in order to draw results from a production environment.

5. Conclusions

In this manuscript, a prefetching scheme for encrypted health data was proposed in order to boost the transmission time of health storage clouds by fetching data as needed and, in general, significantly reduce the delay of the storage cloud. More particularly, the proposed prefetch scheme follows a five-step approach, categorized in an offline and an online phase. During the offline phase, a deep learning network aiming to predict the file that should be prefetched is initially trained using a set of historical data. Afterward, a novel framework based on the influential instances concept is utilized in order to contextualize the results of the deep learning network, while in the online phase, the prefetch scheme performs a prediction of the file that may be requested in an upcoming request and prefetches it. The aforementioned prefetching scheme was evaluated through several experiments with respect to its performance, concluding that there was a substantial decrease in the transmission time when utilized. In addition, tests in order to evaluate the influence of the proposed scheme on the deep learning algorithm were also executed,

confirming that the overall performance of the deep learning network was elevated when combined with the scheme.

Finally, it is within our future plans to further optimize the proposed prefetching scheme in order to reduce the offline phase's learning time, starting with altering the step where the influential instances are identified by discovering them through influence functions [51] or another similar concept. In addition, we aim at completing the integration of the overall proposed storage cloud solution, including the prefetch scheme, within the InteropEHRate project [52], which aims at enabling and supporting citizens within the European Union (EU) and making up new ways to make their health data available when needed. Through the InteropEHRate project, it is within our plans to also evaluate the proposed solution in a real-world scenario, where citizens using their smartphone can back up their health data to a storage cloud such as the one presented in this research while healthcare professionals from trusted healthcare institutions can gain access to this data during emergency situations [53].

In more detail, there is already a scenario under development where the proposed prefetch engine integrated within the storage cloud will be utilized by citizens from various European Union countries (i.e., Italy, Romania, and Belgium) as well as major healthcare institutions such as the Fondazione Toscana Gabriele Monasterio (FTGM) [54] located in Pisa, Italy, the Centre Hospitalier Universitaire de Liège (CHU de Liège) [55] located in Liege, Belgium, as well as the Emergency Hospital Bagdasar-Arseni (SCUBA) [56], a clinical hospital located in Bucharest, Romania. The proposed storage cloud will be deployed and utilized by patients of the abovementioned healthcare institutions along with the healthcare professionals that cure them. Currently, the scenario includes 10 patients from each institution that will utilize the functionalities of the proposed storage cloud and healthcare professionals who, based on a simulated medical emergency, will utilize the Prefetch Engine Component. In addition, a number of encrypted medical resources will be exchanged from and to the proposed storage cloud, including data such as a citizen's allergies and intolerances, chronic conditions and current medications, as well as information about their medical history such as reports of past cardio hospitalizations (discharge reports in PDF format and other structured data). During this scenario, it will be further analyzed how the proposed solution works in a real-world use case, since currently, our work was mainly evaluated in a simulated environment, as has already been described in Section 4.

As a next step, we aim at communicating and disseminating our results in order to enhance their adoption and for them to be integrated by other countries that may still explicitly prohibit health data exchange with third-party entities. Additionally, the fact that the proposed storage cloud is compliant to the General Data Protection Regulation (GDPR) [57] requirements, providing its users the ability to keep the integrity of their data, the "right of access by the data subject (GDPR, Art. 15), the "right to data portability" (GDPR, Art. 20), and the "right to erasure (right to be forgotten)" (GDPR, Art. 17), should contribute to its adoption. Above all, our future plans include further evaluating the proposed architecture with additional datasets and experimental scenarios deployed in the infrastructure of the DIASTEMA project [58], exploiting high-end hardware specifications (e.g., solid-state drives (SSDs) as compared with HDD or DDR4 memories instead of DDR3 with higher capacities and more efficient CPUs, among others) in order for the raw prefetching gain to be more efficiently measured and analyzed, thus not taking under consideration only the download time.

Author Contributions: Conceptualization, C.S., G.M., A.K. and D.K.; methodology, C.S., A.K., A.M. and G.M.; software, C.S., A.K. and G.M.; validation, C.S., A.K. and G.M.; formal analysis, C.S., A.K. and A.M.; investigation, C.S., A.K. and G.M.; resources, C.S. and G.M.; data curation, C.S. and G.M.; writing—original draft preparation, C.S.; writing—review and editing, C.S., A.K., A.M., G.M. and D.K.; visualization, C.S. and G.M.; supervision, D.K.; project administration, D.K.; funding acquisition, D.K. All authors have read and agreed to the published version of the manuscript.

Funding: The research leading to these results received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 826106 (InteropEHRate project) and the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation under the call RESEARCH—CREATE—INNOVATE (project code: DIASTEMA—T2EDK-04612).

Data Availability Statement: Not applicable, the study does not report any data.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Maufe, Z. Financial Services, Cloud Adoption, Regulators. Google Cloud. 12 August 2021. Available online: https://cloud.google.com/blog/topics/inside-google-cloud/new-study-shows-cloud-adoption-increasing-in-financial-services (accessed on 19 January 2022).
2. Ezell, S.B.S. How Cloud Computing Enables Modern Manufacturing. Information Technology & Innovation Foundation (ITIF). 22 June 2017. Available online: https://itif.org/publications/2017/06/22/how-cloud-computing-enables-modern-manufacturing (accessed on 19 January 2022).
3. Bai, C.; Dallasega, P.; Orzes, G.; Sarkis, J. Industry 4.0 technologies assessment: A sustainability perspective. *Int. J. Prod. Econ.* **2020**, *229*, 107776. [CrossRef]
4. Sonin, J.; Lakey Becker, A.; Nipp, K. It's Time for Individuals—Not Doctors or Companies—To Own Their Health Data. STAT. 12 November 2021. Available online: https://www.statnews.com/2021/11/15/its-time-for-individuals-not-doctors-or-companies-to-own-their-health-data/ (accessed on 19 January 2022).
5. McCurry, C. People Should Have Ownership of Personal Health Data, Says Patients' Group. Independent. 7 September 2021. Available online: https://www.independent.ie/breaking-news/irish-news/people-should-have-ownership-of-personal-health-data-says-patients-group-40824715.html (accessed on 19 January 2022).
6. Google Health. Google Health. 2021. Available online: https://health.google/ (accessed on 19 January 2022).
7. Microsoft. Cloud for Healthcare. 2021. Available online: https://www.microsoft.com/en-us/industry/health/microsoft-cloud-for-healthcare (accessed on 19 January 2022).
8. Estonian Central Health Information System and Patient Portal. CEF Digital. 26 July 2019. Available online: https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/2019/07/26/Estonian+Central+Health+Information+System+and+Patient+Portal (accessed on 19 January 2022).
9. IBM. Watson Health Citizen Engagement–Details. 2021. Available online: https://www.ibm.com/products/citizen-engagement/details (accessed on 19 January 2022).
10. Amazon Web Services, Inc. AWS for Health. 2021. Available online: https://aws.amazon.com/health/ (accessed on 19 January 2022).
11. Health Cloud. Salesforce. 2021. Available online: https://www.salesforce.com/eu/products/health-cloud/overview/ (accessed on 19 January 2022).
12. Esteves, S.; Silva, J.N.; Veiga, L. Palpatine: Mining Frequent Sequences for Data Prefetching in NoSQL Distributed Key-Value Stores. In Proceedings of the 2020 IEEE 19th International Symposium on Network Computing and Applications (NCA), Cambridge, MA, USA, 24–27 November 2020; pp. 1–10.
13. Chen, Y.; Zhang, Y.; Wu, J.; Wang, J.; Xing, C. Revisiting Data Prefetching for Database Systems with Machine Learning Techniques. In Proceedings of the 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 19–22 April 2021; pp. 2165–2170.
14. Singh, T.B.; Chitra, S. Prefetching of Web Objects For Effective Retrieval Process Through Data Mining Techniques. 2021. Available online: https://assets.researchsquare.com/files/rs-266666/v1/ef3494ab-9c7c-4cb1-979e-4d7ea8a465d9.pdf?c=1631881773 (accessed on 19 January 2022).
15. Hussien, N.; Sulaiman, S. Web pre-fetching schemes using machine learning for mobile cloud computing. *Int. J. Adv. Soft Comput. Appl.* **2017**, *9*, 154–187.
16. Konstantinidis, A.; Irakleous, P.; Georgiou, Z.; Zeinalipour-Yazti, D.; Chrysanthis, P.K. IoT data prefetching in indoor navigation SOAs. *ACM Trans. Internet Technol.* **2018**, *19*, 1–21. [CrossRef]
17. Shakarami, A.; Ghobaei-Arani, M.; Shahidinejad, A.; Masdari, M.; Shakarami, H. Data replication schemes in cloud computing: A survey. *Clust. Comput.* **2021**, *24*, 2545–2579. [CrossRef]
18. Mansouri, N.; Javidi, M.M. A new prefetching-aware data replication to decrease access latency in cloud environment. *J. Syst. Softw.* **2018**, *144*, 197–215. [CrossRef]
19. Liang, Y.; Hu, Z.; Zhang, X.; Xiao, H. Correlation-aware replica prefetching strategy to decrease access latency in edge cloud. *China Commun.* **2021**, *18*, 249–264. [CrossRef]
20. Alyami, M.A.; Almotairi, M.; Aikins, L.; Yataco, A.R.; Song, Y.T. Managing personal health records using meta-data and cloud storage. In Proceedings of the 2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS), Wuhan, China, 24–26 May 2017; pp. 265–271.

21. Patra, D.; Ray, S.; Mukhopadhyay, J.; Majumdar, B.; Majumdar, A.K. Achieving e-health care in a distributed EHR system. In Proceedings of the 2009 11th International Conference on e-Health Networking, Applications and Services (Healthcom), Sydney, Australia, 16–18 December 2009; pp. 101–107.
22. Sun, J.; Fang, Y. Cross-domain data sharing in distributed electronic health record systems. *IEEE Trans. Parallel Distrib. Syst.* **2009**, *21*, 754–764.
23. Bahga, A.; Madisetti, V.K. A cloud-based approach for interoperable electronic health records (EHRs). *IEEE J. Biomed. Health Inform.* **2013**, *17*, 894–906. [CrossRef] [PubMed]
24. Jin, Y.; Deyu, T.; Yi, Z. A distributed storage model for EHR based on Hbase. In Proceedings of the 2011 International Conference on Information Management, Innovation Management and Industrial Engineering, Shenzhen, China, 26–27 November 2011; Volume 2, pp. 369–372.
25. Apache Hbase. Apache Hbase. 2021. Available online: https://hbase.apache.org/ (accessed on 19 January 2022).
26. Apache Hadoop. Apache Hadoop. 2021. Available online: https://hadoop.apache.org/ (accessed on 19 January 2022).
27. Zheng, X.; Mukkamala, R.R.; Vatrapu, R.; Ordieres-Mere, J. Blockchain-based personal health data sharing system using cloud storage. In Proceedings of the 2018 IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom), Ostrava, Czech Republic, 17–20 September 2018; pp. 1–6.
28. Song, Y.T.; Hong, S.; Pak, J. Empowering patients using cloud based personal health record system. In Proceedings of the 2015 IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), Takamatsu, Japan, 1–3 June 2015; pp. 1–6.
29. Karim, F.; Majumdar, S.; Darabi, H.; Chen, S. LSTM fully convolutional networks for time series classification. *IEEE Access* **2017**, *6*, 1662–1669. [CrossRef]
30. Webby, R.; O'Connor, M. Judgemental and statistical time series forecasting: A review of the literature. *Int. J. Forecast.* **1996**, *12*, 91–118. [CrossRef]
31. Ho, S.L.; Xie, M. The use of ARIMA models for reliability forecasting and analysis. *Comput. Ind. Eng.* **1998**, *35*, 213–216. [CrossRef]
32. Zhang, J.; Man, K.F. Time series prediction using RNN in multi-dimension embedding phase space. In Proceedings of the SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218), San Diego, CA, USA, 14 October 1998; Volume 2, pp. 1868–1873.
33. Gers, F.A.; Eck, D.; Schmidhuber, J. Applying LSTM to time series predictable through time-window approaches. In *Neural Nets WIRN Vietri-01*; Springer: London, UK, 2002; pp. 193–200.
34. Hochreiter, S.; Bengio, Y.; Frasconi, P.; Schmidhuber, J. Gradient Flow in Recurrent Nets: The Difficulty of Learning Long-Term Dependencies. 2001. Available online: https://www.researchgate.net/profile/Y-Bengio/publication/2839938_Gradient_Flow_in_Recurrent_Nets_the_Difficulty_of_Learning_Long-Term_Dependencies/links/546cd26e0cf2193b94c577c2/Gradient-Flow-in-Recurrent-Nets-the-Difficulty-of-Learning-Long-Term-Dependencies.pdf (accessed on 19 January 2022).
35. Kwon, B.C.; Choi, M.J.; Kim, J.T.; Choi, E.; Kim, Y.B.; Kwon, S.; Sun, J.; Choo, J. Retainvis: Visual analytics with interpretable and interactive recurrent neural networks on electronic medical records. *IEEE Trans. Vis. Comput. Graph.* **2018**, *25*, 299–309. [CrossRef] [PubMed]
36. Adadi, A.; Berrada, M. Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access* **2018**, *6*, 52138–52160. [CrossRef]
37. Huang, X.; Ye, Y.; Xiong, L.; Lau, R.Y.; Jiang, N.; Wang, S. Time series k-means: A new k-means type smooth subspace clustering for time series data. *Inf. Sci.* **2016**, *367*, 1–13. [CrossRef]
38. Senin, P. *Dynamic Time Warping Algorithm Review*; Information and Computer Science Department University of Hawaii at Manoa: Manoa, HI, USA,, 2008; Volume 855, p. 40.
39. Symvoulidis, C.; Kiourtis, A.; Mavrogiorgou, A.; Kyriazis, D. Healthcare Provision in the Cloud: An EHR Object Store-based Cloud Used for Emergency. In Proceedings of the HEALTHINF, Online Streaming, 11–13 February 2021; pp. 435–442.
40. Symvoulidis, C.; Mavrogiorgou, A.; Kiourtis, A.; Marinos, G.; Kyriazis, D. Facilitating Health Information Exchange in Medical Emergencies, In Proceedings of the 2021 International Conference on e-Health and Bioengineering (EHB), Iasi, Romania, 18–19 November 2021.
41. MinIO, Inc. MinIO | High Performance, Kubernetes Native Object Storage. *MinIO*. 2021. Available online: https://min.io/ (accessed on 19 January 2022).
42. Pramukantoro, E.S.; Bakhtiar, F.A.; Bhawiyuga, A. A Semantic RESTful API for Heterogeneous IoT Data Storage. In Proceedings of the 2019 IEEE 1st Global Conference on Life Sciences and Technologies (LifeTech), Osaka, Japan, 12–14 March 2019; pp. 263–264.
43. Goli-Malekabadi, Z.; Sargolzaei-Javan, M.; Akbari, M.K. An effective model for store and retrieve big health data in cloud computing. *Comput. Methods Programs Biomed.* **2016**, *132*, 75–82. [CrossRef] [PubMed]
44. Pandey, M.K.; Subbiah, K. A novel storage architecture for facilitating efficient analytics of health informatics big data in cloud. In Proceedings of the 2016 IEEE International Conference on Computer and Information Technology (CIT), Nadi, Fiji, 8–10 December 2016; pp. 578–585.
45. MongoDB. MongoDB: 2021. The Application Data Platform. Available online: https://www.mongodb.com/ (accessed on 19 January 2022).
46. Flask. Flask. 2021. Available online: https://flask.palletsprojects.com/en/2.0.x/ (accessed on 19 January 2022).
47. FHIR. Patient. 2021. Available online: https://www.hl7.org/fhir/patient.html (accessed on 19 January 2022).

48. FHIR. Bundle. 2021. Available online: https://www.hl7.org/fhir/bundle.html (accessed on 19 January 2022).
49. FHIR. Composition. Available online: https://www.hl7.org/fhir/composition.html (accessed on 19 January 2022).
50. DICOM. 2022. Available online: https://www.dicomstandard.org/ (accessed on 19 January 2022).
51. Koh, P.W.; Liang, P. Understanding black-box predictions via influence functions. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 1885–1894.
52. InteropEHRate. InteropEHRate. 2019. Available online: https://www.interopehrate.eu/ (accessed on 19 January 2022).
53. Koutsoukos, K.; Symvoulidis, C.; Kiourtis, A.; Mavrogiorgou, A.; Dimopoulou, S.; Kyriazis, D. Emergency Health Protocols Supporting Health Data Exchange, Cloud Storage, and Indexing. In Proceedings of the 15th International Conference on Health Informatics-HEALTHINF 2022, Vienna, Austria, 9–11 February 2022; pp. 597–604.
54. Fondazione Toscana Gabriele Monasterio. Fondazione Toscana Gabriele Monasterio. 2022. Available online: https://www.monasterio.it (accessed on 19 January 2022).
55. CHU de Liège. CHU de Liège. 2022. Available online: https://www.chuliege.be (accessed on 19 January 2022).
56. Emergency Hospital Bagdasar-Arseni. Emergency Hospital Bagdasar-Arseni. 2022. Available online: https://www.bagdasar-arseni.ro (accessed on 19 January 2022).
57. General Data Protection Regulation (GDPR) Compliance Guidelines Proton Technologies AG. 2022. Available online: https://gdpr.eu/ (accessed on 19 January 2022).
58. Diastema Diastema. 2022. Available online: https://diastema.gr/ (accessed on 19 January 2022).