Prediction of Mobile App Privacy Preferences with User Profiles via Federated Learning

André Brandão
CRACS/INESCTEC, CISUC and Dep. of Computer Science, Faculty of Sciences, University of Porto
Porto, Portugal
andrebrandao@ua.pt

Ricardo Mendes
CISUC, Dep. of Informatics Engineering, University of Coimbra
Coimbra, Portugal
rscmendes@dei.uc.pt

João P. Vilela
CRACS/INESCTEC, CISUC and Dep. of Computer Science, Faculty of Sciences, University of Porto
Porto, Portugal
jvilela@fc.up.pt

ABSTRACT

Permission managers in mobile devices allow users to control permissions requests, by granting or denying application's access to data and sensors. However, existing managers are ineffective at both protecting and warning users of the privacy risks of their permissions' decisions. Recent research proposes privacy protection mechanisms through user profiles to automate privacy decisions, taking personal privacy preferences into consideration. While promising, these proposals usually resort to a centralized server towards training the automation model, thus requiring users to trust this central entity. In this paper we propose a methodology to build privacy profiles and train neural networks for prediction of privacy decisions, while guaranteeing user privacy, even against a centralized server. Specifically, we resort to privacy-preserving clustering techniques towards building the privacy profiles, that is, the server computes the centroids (profiles) without access to the underlying data. Then, using federated learning, the model to predict permission decisions is learnt in a distributed fashion while all data remains locally in the users' devices. Experiments following our methodology show the feasibility of building a personalized and automated permission manager guaranteeing user privacy, while also reaching a performance comparable to the centralized state of the art, with an F1-score of 0.9.

CCS CONCEPTS

• **Security and privacy → Privacy-preserving protocols**; **Domain-specific security and privacy architectures**.

KEYWORDS

Privacy, mobile devices, automated prediction, federated learning

ACM Reference Format:
André Brandão, Ricardo Mendes, and João P. Vilela. 2022. Prediction of Mobile App Privacy Preferences with User Profiles via Federated Learning. In *Proceedings of the Twelveth ACM Conference on Data and Application Security and Privacy (CODASPY '22), April 24–27, 2022, Baltimore, MD, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3508398.3511526

CODASPY '22, April 24–27, 2022, Baltimore, MD, USA.

ACM ISBN 978-1-4503-9220-4/22/04…$15.00
https://doi.org/10.1145/3508398.3511526

1 INTRODUCTION

Privacy is recognized by the United Nations' Declaration of Human Rights of 1948 [23] as a fundamental human right. Alan Westin's noted in his 1967 book *Privacy and Freedom* "a deep concern over the preservation of privacy under the new pressures from surveillance technology". More than 50 years have passed, and with the emergence of smartphones, that concern is now more present then ever.

Smartphones are mobile devices that gather almost the same capabilities as personal computers. Besides the traditional voice calls and text messages, they contain multimedia functionalities like music, image, video or gaming. They incorporate a variety of sensors like magnetometer, gyroscope, accelerometer or proximity sensors and support communication protocols such as Wi-Fi, Bluetooth or satellite navigation, which allows the collection of a variety of data [4]. Current modern smartphones are able to collect vast amounts of information like location, photos, messages, call log, contacts or emails. On top of this, it is possible to extract high level information from this data, like home address, work address or close friends. Due to the widespread adoption of these devices, and the ease of developing applications for smartphones, the barrier to collect personal information from the masses is now much lower. This led to privacy concerns, specially after events where the data collected was used for malicious purposes, like the Cambridge Analytica scandal [8].

All this data collection is done by the apps installed on users smartphones. Users can reduce the amount of data shared with the applications by allowing or denying specific permissions (e.g. location of camera) for each application. To allow for user control over these permissions, smartphones implement two permission systems: Ask-On-Install (AOI) and Ask-On-First-Use (AOFU), the first asks the user to define the permissions when the application is installed and the latter when it is first used. In the AOFU strategy, the user can always change the permissions configuration on the phone settings later on, a practice that is mostly unused [2].

There are problems with both AOI and AOFU strategies. With AOI, either the user allows all requested permissions at the installation prompt or has to refuse the installation of the application. Regrettably, people do not pay attention or fail to understand the prompts [5]. Furthermore, these static permissions do not account for the user context, such as the user location, if the application asking the permission is being used or not, or the time of the day [5, 6, 10, 25].

In the AOFU system, every time an application asks for a permission for the first time, the user has to choose to allow or deny

it. This allows for selective permission control, thus granting fine-grained permission control [2]. Additionally, this decision will be based on that specific context, as the prompt is contextualized by the runtime need for the resource, which makes it better than the AOI system [2, 21]. However, after being allowed a permission once, the app will have this permission automatically granted for all subsequent uses, even without the user noticing. Therefore, AOFU does not account for the context when automatically granting permissions.

Taking into consideration what was said above, there is still a gap in the research on what influences users' decisions in terms of app permissions. The current model (AOFU) does not represent how the users think about privacy. According to the data collected by our campaign, each user has on average about 35 permission requests per hour, which makes manual answering unfeasible. So, there is a need to create tools to automate these decisions, that are capable to represent the user intentions as a function of the context of the user and of the context of the device.

Current research proposals reduce the burden originating from permission prompts by automating the user decisions [12, 13, 17]. However, some of these mechanisms are trained locally, which requires intensive user input, thus leading to user fatigue, while others rely on a centralized approach, where all the users' privacy preferences data must be sent to a central server.

In this paper we propose a system capable of learning the users' privacy preferences according to the context with privacy guarantees, *i.e.* the users' privacy decisions and contextual information is not disclosed, even to server training the model. Towards this end, the system generates privacy-profiles in a secure way, using privacy preserving distributed hierarchical clustering and efficient privacy preserving distributed k-means for non-IID data. These profiles allow for the personalization of the predictive model, while reducing the amount of input required from the user [11]. Finally, using these profiles, the system trains a model in a secure way, using federated learning, to predict the users' responses to the permissions requests. We evaluate the feasibility of this system and compare its performance to the non-private centralized approach, using a real world dataset. Our results demonstrate that the proposed system is able to achieve a performance comparable to a centralized non-private approach, that is on par with the state-of-the-art for privacy decisions prediction.

The rest of the paper is organized as follows. Section 2 provides a brief overview of the related work done in privacy preferences automation. In Section 3 we describe the dataset that we used to evaluate our system and the process to generate privacy profiles. Section 4 presents our strategy to predict users privacy preferences using federated learning. In Section 5 we present the results from the validation and testing of our strategies. Section 6 presents potential future work, and finally, Section 7 draws the concluding remarks.

2 RELATED WORK

In order to enhance permission managers, researchers proposed the use of automation systems that either predict or recommend permission settings, while taking into consideration personal preferences. In order to reduce the amount of input required from a user, these preferences are typically captured through privacy profiles. However, several user profiling techniques can be used to generate these profiles.

Agarwal and Hall [1] developed **ProtectMyPrivacy**, a system for iOS devices that recommends privacy decisions to users based on crowdsourcing. All users send their privacy decisions to a centralized server, then the "experts", about 1% of the most active users, contribute to the recommendation system. This approach does not protect user privacy and the system only provides universal recommendations, *i.e.* only one profile exists and the same recommendation is sent to every user. This approach does not take into account each users' individual characteristics or the context, instead it assumes that a consensus will emerge from the crowd. Rashidi, Fung, and Vu [19] also developed a crowdsourcing system named **RecDroid**. Despite having a more advanced system for recommending privacy decisions than **ProtectMyPrivacy**, RecDroid suffers from the exact same problems. Users' privacy decisions are sent to the server, exposing their data to anyone who has access to the server, in a lawful or unlawful way, and only one profile exists for every user.

Zhao, Ye, and Henderson [27] proposed collaborative filtering strategies to recommend privacy settings for location permissions. However, the data collection was done using surveys, where specific scenarios where presented to the participants and they had to select a privacy setting. It is hard to understand if this simulation is representative of real life decisions as aspirational responses often diverge from real behavior [16].

Xie, Knijnenburg, and Jin [26] also proposed a location-privacy recommender system based on collaborative filtering. The data, collected by 40 volunteers in University of St. Andrews, is only related to the location status on the Facebook[1] app.

Ismail et al. [9] conducted a user study with 26 participants to create a collaborative-filtering recommender system for Instagram[2] privacy settings. They also used K-nearest neighbours to create profiles with similar users, from which the collaborative-filtering will recommend the permission setting.

Lin *et al.* [11] used Amazon Mechanical Turk to collect privacy preferences with regard to over 800 apps from over 700 participants. The data collected corresponded to the privacy decision for each (permission, purpose) tuple. They applied hierarchical clustering over their dataset and identified four distinct clusters. With the results obtained, they concluded that the profiles were capable of predicting many of a user's mobile app privacy preferences. Liu *et al.* [13] collected privacy preferences from the LBE Privacy Guard[3] application. The collected data from users mainly based in mainland China, corresponds to over 239000 users and 12119 apps, resulting in a total of 28630179 decisions. A problem with their dataset is that this data was collected from users who already had rooted phones, so their dataset is very biased since all participants are all tech savvy. They used k-Means, with each user represented by one vector with every (app, permission, decision) tuple combination, to generated six privacy profiles. With the resulting profiles they were capable of improving the users' permission preferences predictions.

Ravichandran *et al.* [20] generated profiles for location-sharing applications, where users have to choose if they are willing to let

[1] www.facebook.com (2021, August 17)
[2] www.instagram.com (2021, August 17)
[3] http://www.lbesec.com (2021, August 17)

others see their locations under specific conditions. The data used in the project was collected in 2008 from 30 users over a period of one week. In order to generate the profiles, they represented each user by training a decision tree to extract policies. With each user represented as a vector of policies, K-means was then used to cluster those policies in profiles. The authors concluded that the resulting profiles were capable of improving the predicting accuracy of the users' choices, in comparison to the model without the profiles.

Sanchez *et al.* developed a privacy-settings recommendation system for fitness devices [22]. Using Amazon Mechanical Turk, they recruited 310 participants to answer a survey. They collected data related to demographics, phone permissions, the type of data collected by the sensors and the entity to which the data is going to be shared with. They proceeded to apply the K-Modes algorithm, similar to the k-Means but more suitable to nominal variables, where instead of using the mean to compute the centroid, the mode is used. With the resulting 6 profiles, where each user can belong to more than one profile, they conclude that the profiles helped the recommendation system to make better decisions.

Liu *et al.* [12] collected permission settings from 84 Android users around the world. Representing user data as a vector of decisions for every combination of the (app category, permission, purpose) tuple, they applied hierarchical clustering to obtain 7 privacy profiles. These profiles were then used as input in a Support Vector Machine (SVM) classifier to predict user preferences to apps' future permission requests. Using profiles as input in the SVM classifier improved the F1-score of the model from 74.24% to 90.02%.

From the reviewed literature we conclude that the strategies for generating privacy profiles can be divided in two categories: hierarchical clustering and k-means, where the user is represented by a single point or multiple points. However, all proposed methodologies rely on a centralized server to create the profiles using the privacy preferences of the users. This paradigm can compromise user privacy, specially in methodologies that use contextual features [17], such as the location of the user.

In this paper, our goal is to generate these profiles while preserving user privacy even against the centralized server. Therefore, in the next sections we introduce methods to generate privacy profiles in a secure way (section 3), and then in section 4, we present a strategy to privately predict the users' privacy preferences using federated learning with the generated profiles as features.

3 SECURE GENERATION OF PRIVACY PROFILES

The creation of privacy profiles has traditionally resorted to the use of a centralized server that receives personal privacy decisions and the surrounding contextual data, such as the requesting application or even the user location. In this section we use a real world dataset, which we describe and analyze in section 3.1, to demonstrate and compare the generation of privacy profiles using secure and non-secure algorithms in section 3.2. Section 4 then presents an approach to train a model with these profiles towards predicting privacy preferences in a privacy-preserving way.

3.1 Data Characterization Overview

In this work we used the dataset collected in the NGI Trust project COP-MODE[4] through COP-MODE's Naive Permission Manager[5] that we developed to intercept permission requests and prompt the user for the following data: answer to the request (allow or deny), the semantic location and if it was expecting that request to appear. At the time of the prompt, the app also collects contextual data regarding the phone state and the user context, which includes information such as background and foreground running apps, network status, geographic coordinates, semantic location, and devices in neighborhood. This research project was approved by the ethics committees of the Faculty of Sciences of the University of Porto, Portugal, and of the University of Cambridge, U.K. A complete description and analysis of the data collected can be found in Mendes *et al.* [15] and in the COP-MODE project's webpage[4].

The data was collected through several campaigns occurring between end of July 2020 and end of March 2021 in Portugal. The dataset comprises 93 users and 2180302 permissions, from which 65261 were manually answered by the users, with an average of 701.73 answered requests per user. From the 65261 permissions, 66% were accepted and 33% denied.

Since some campaigns were done during the COVID-19 confinement, more than 80% of the permissions were asked when the users were at home, 6.8% at work, 6.3% while travelling and the remaining at "other locations". At work and while travelling, users tend to allow much more permissions, unlike at "other locations", where the users deny almost as many permissions as they grant. At home, users tend to grant permissions twice as much as they deny.

Currently, the Android operating system contains 12 permission groups: `CALL_LOG`, `SMS`, `CALENDAR`, `CAMERA`, `CONTACTS`, `LOCATION`, `MICROPHONE`, `PHONE`, `SENSORS`, `ACTIVITY_RECOGNITION`, `NEARBY_DEVICES` and `STORAGE`. These permission groups comprise a set of permissions, for example, the `SMS` group permission encompasses the permissions `READ_SMS` and `SEND_SMS`. Because the Android default permission manager only requests permissions at the group level, we focus our analysis on permission groups and refer to these simply as permissions. There are two main permissions groups, `LOCATION` and `CONTACTS`, accounting for 50% of the requests. `PHONE` and `STORAGE` permissions account for 37% of the requests.

Figure 3 gives us an overall view of the users tendencies to allow and deny specific app categories and permissions. The app categories were extracted from the Google Play Store[6], from which `MUSIC_AND_AUDIO`, `ENTERTAINMENT`, `SPORTS`, `SHOPPING`, and `GAME` were merged together in the `ENTERTAINMENT` category. For each combination of app category and permission, it is represented the average decision of all users by the color of the cell, where dark green corresponds to all those requests being accepted (average of 1), and the dark red represents all permission denied (average of -1). For example, almost all requests for `CALENDAR` from `ENTERTAINMENT` apps were accepted by the users, and almost all requests for `PHONE` from `PHOTOGRAPHY` apps were denied.

From Figure 3 we can observe that the most denied permissions are `MICROPHONE` and `PHONE` and the most denied app categories

[4]https://cop-mode.dei.uc.pt/dataset (2021, October 14)
[5]https://cop-mode.dei.uc.pt/cm-npm (2021, October 14)
[6]https://play.google.com/store (2021, August 17)

App category:	EVENTS	EVENTS	...	AUTO_AND_VEHICLES	AUTO_AND_VEHICLES
Requested permission:	CALENDAR	CAMERA	...	PHONE	CONTACTS
Grant result:	0.9	0	...	0	0
	0.2	0.1	...	0.35	0.4
			⋮		
	0.6	0.2	...	0.15	0.2

Table 1: Vector representation of privacy preferences per pair of app category, requested permission, where each row represents one user. Each value represents the normalized grant result for each pair in the interval between 0, to 1, where 0 corresponds to denying all requests and 1 allowing all.

are TOOLS, FINANCE, VIDEO_PLAYERS, NEWS_AND_MAGAZINES. We also conclude that the most accepted permissions are STORAGE, CALENDAR and CAMERA and the most accepted app categories are ART_AND_DESIGN, MEDICAL and AUTO_AND_VEHICLES. Furthermore, both LOCATION and CONTACTS permissions have many yellow/orange squares. This indicates that users are not in agreement with each other and/or there are more variables that makes the same user accept and deny the same permission in different contexts, like its current location for example.

3.2 Privacy Profiles

As discussed in section 2 there are many possible methods to generate profiles. In this section, we design strategies to generate privacy profiles in a privacy-preserving manner with the COP-MODE project data, resorting to hierarchical clustering (section 3.2.1) and k-Means (section 3.2.3).

3.2.1 Hierarchical Clustering. One of the possibility towards the generation of user profiles is using the hierarchical clustering algorithm. In order to apply this strategy to our dataset, for each user answered permission we group the data by `userID`, `category` and `permission`, with the average of the `grantResult`. So, for each user, we have the average `grantResult` for every combination of `category` and `permission`. However, most users do not have `grantResults` for all these possible combinations. So, these data points are filled by a multivariate imputer, where the `grantResult` is modeled as a polynomial function of all the remaining features. We chose this imputation strategy for being more sophisticated than univariate imputation strategies. We used the `IterativeImputer` class from the Scikit-Learn package [18].

After the imputation, all the `category`, `permission` tuples are flattened in the same row as illustrated in table 1, *i.e.* a user is a row with one column for every `category`, `permission` combination, where its value is the average `grantResult`. With the resulting matrix, we can build a dendrogram (figure 5) and if we cut the dendogram horizontally at distance equal to 4.3 with observe three clusters of users: orange; green and red; purple, brown, blue and rose (figure 4).

For demonstration purposes, we created $k = 3$ clusters, which we illustrate in the form of the average `grantResult` of the users in the cluster for each category, permission combination. Figures 4a, 4b and 4c presents these profiles. We can observe that users in profile 1 (figure 4a) tend to deny most of the permissions on most of the categories, in profile 2 (figure 4b), the opposite happens as users accept almost always all permissions related to all categories, with the exception of (CAMERA, PHOTOGRAPHY), (MICROPHONE, FINANCE) and (PHONE, NEWS_AND_MAGAZINES). Finally, in profile 3 (figure 4c), we observe less consistent behaviour, with many yellow/orange rectangles, meaning the users allow and deny that permission/category combinations more or less the same number of times.

3.2.2 Privacy Preserving Distributed Hierarchical Clustering. To generate privacy profiles using hierarchical clustering with privacy guarantees, we can use the privacy preserving distributed hierarchical clustering algorithm [7]. This algorithm allows the construction of any agglomerative hierarchical clustering algorithm over horizontally partitioned data. The strategy is based on the secure scalar product, an algorithm proposed by Vaidya and Clifton [24], that allows the computation of the scalar product by using linear combinations of random numbers to make vector elements, and then apply some computations to eliminate the effect of the random numbers from the result. Using the following equation to obtain the distance between point X and point Y: $(X \cdot X + Y \cdot Y - 2X \cdot Y)^{\frac{1}{2}}$, we can obtain the euclidean distance between two points (needed for clustering) in a secure way, without sharing either point X or Y.

In this context, each user only has access to their own local dataset. So, the process of flattening and filling of missing data referred in section 3.2.1 needs to be applied independently to each local dataset. This can cause problems in the imputation step if our imputation algorithm needs access to all the users' data. So, our imputation strategy must rely solely on the local data or global statistics that can be acquired using the secure aggregation strategy. Therefore, we perform the imputation with the `IterativeImputer` class from the Scikit-Learn Python package [18]. Each user local data thus consist of a single row/vector that is then used to generate the profiles using the privacy preserving distributed hierarchical clustering algorithm.

3.2.3 k-Means. One strategy capable of generating/assigning multiple profiles for each user is k-Means [22]. To apply this algorithm to our data, we group the data by `userID`, `category` and `permission`, with the average of the `grantResult`, for each user, similarly to the process taken for the hierarchical clustering in section 3.2.1. However, this time, each user is represented by multiple rows of `category`, `permission` and `grantResult` as illustrated in table 2. Unlike the strategy in section 3.2.1, we do not need to flatten the user data in one row, so we also do not need to impute the missing combinations of `category` and `permission`. This is a clear advantage of this method, since it removes the added bias from the imputation. We can feed this data directly to the k-Means

algorithm and generate the privacy profiles, where each user will have data points in one or more profiles.

Since users can have data points in multiple profiles, the profile representation is less intuitive than in the hierarchical clustering approach. In order to represent the users and keep as much information as possible, we decided to associate a percentage of each profile to every user. For example, if a user has 10 data points in profile 1, 30 in profile 2 and 60 in profile 3 on a total of 3 profiles, this user will be represented as $[0.1, 0.3, 0.6]$, instead of being represented as $[0, 0, 1]$, for instance. This representation is needed in order to use the profile's information to predict the `grantResult` in section 4.

UserID	Permission	Category	Avg. Grant Result
1	FINANCE	CALENDAR	0.90
2	FINANCE	CALENDAR	0.20
1	FINANCE	CAMERA	0.00
2	FINANCE	CAMERA	0.10
	⋮		
1	SOCIAL	CONTACTS	0.00
2	SOCIAL	CONTACTS	0.40

Table 2: Matrix representation of privacy preferences, where one user is represented by one or more rows. Each value represents the normalized grant result in the interval between 0, to 1, where 0 corresponds to denying all requests and 1 allowing all requests for the given user, permission and category in the row.

3.2.4 Efficient Privacy Preserving Distributed k-Means. To generate privacy profiles with privacy guarantees, using the k-Means algorithm, we can use the efficient privacy preserving distributed k-Means algorithm [3] as it is efficient and robust to non-IID data. The base idea of this approach consists in each client computing the k-Means algorithm locally, with a variable number of clusters. The server will use the resultant centroids to apply the K-Means algorithm again, discovering the global centroids. To maintain the client's privacy, homomorphic encryption and secure aggregation is used in the process of learning the global centroids, such that the server only sees encrypted data, therefore preserving the privacy of the clients. In the end, each user will have the association between every local data point and the respective profile. With this information, the user can extract the profile representation as described in section 3.2.3. Since only the local centroids are sent to the server and used to find the global centroids, this algorithm is efficient and reduces transmission costs, thus being suitable in a real world scenario.

4 FEDERATED LEARNING FOR GRANT PREDICTION

In this section we describe how to use federated learning [14] to predict the users' answers to a permission request, while preserving user privacy. Federate learning provides us that possibility by training a neural network model locally, on each smartphone, using only local data, and then sharing only the neural network weights with a central server on each iteration. The central server averages the weights and returns the result to the clients, so they can use these new weights to continue the training process.

To use the data we collected from the users as input in the machine learning model, we scale the data using `MinMaxScaler`, which scales the data points to a range between 0 and 1. We also applied one-hot encoding to the dataset, so each categorical variable is represented by a vector of 0s and 1s. The following features are used as input of the neural network to predict the `grantResult`:

- `app_category` – the category of the app as retrieved from the Google Play Store[7];
- `checkedPermissionGroup` – the permission group of the requested permission – categorical variable;
- `checkedPermission` – the requested permission – categorical variable;
- `isTopAppRequestingApp` – whether the requesting app is the app being shown to the user – Binary variable;
- `screenIsInteractive` – whether the screen is on and the phone unlocked (interactive mode) – Binary variable;
- `method` – the function used to request the permission – categorical variable;
- `hour` – hour of the day – categorical variable;
- `weekday` – the day of the week – categorical variable;
- `isForeground` – whether the requesting app is in the foreground – binary variable;
- `networkStatus` – the type of network connection (disconnected, mobile network or wi-fi) – categorical variable;
- `profile` – representation of the privacy profile – categorical variable.

These are the non-unique features collected by the permission manager, *i.e.* the features that have repeating values, unlike `ID` like features. The `selectedSemanticLoc` and `wasRequestExpected` were removed, since they require user interaction. The timestamp was transformed into hour and weekday, since the timestamp by itself is unique.

The `profile` feature is represented as a one-hot encoding vector, in the profiles generated by the hierarchical clustering algorithms of the previous sections 3.2.1 and 3.2.2. For example, a user in profile 3 of a total of 4 profiles would be represented as the user 1 in table 3. In case the profiles are generated by the k-Means algorithm, the profiles are represented as percentage-wise, where we still have a column for each profile, but instead of 0s and 1s we have the percentage of data points per profile (user 2 in table 3).

User	Profile 1	Profile 2	Profile 3	Profile 4
user 1	0	0	1	0
user 2	0.1	0	0.7	0.2

Table 3: Possible representations for the profile feature.

Figure 6 represents the entire process in a diagram. The first step of this strategy is to generate the privacy profiles using one of the methods described in section 3.2. The output of the clustering algorithm, that is, the profiles are then added to the local dataset, *i.e.* for each row in the dataset, we add the respective profile ID. Then,

[7] https://support.google.com/googleplay/android-developer/answer/9859673 (2021, August 17)

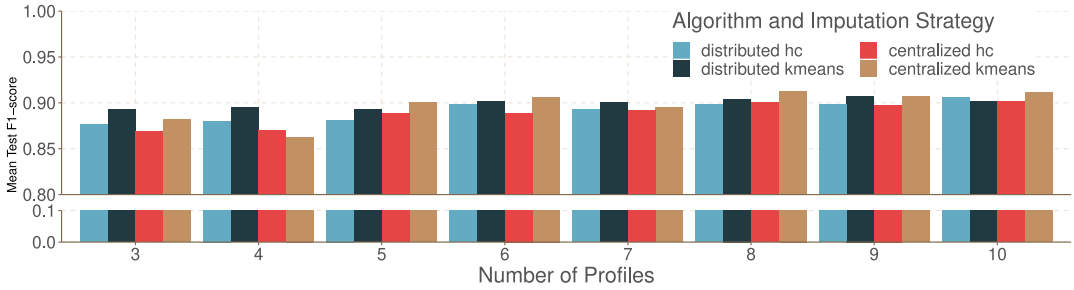


Figure 1: Mean F1-score 5-fold cross validation results for the prediction of the grant result with different clustering strategies for generation of privacy profiles: centralized and distributed versions for both hierarchical clustering (hc) and k-means clustering.

one-hot encoding is applied to the dataset, and a division of 66% for training and 33% for testing is applied. In order to reduce the bias of the model we oversample each local training dataset after dividing it, obtaining 50% training points with granted permissions, and 50% with denied permissions. This way, the trained model will be less biased, yet the test set will have a realistic percentage of granted permissions, since it is not oversampled.

Now, the centralized server creates a Neural Network and shares the weights with all the clients. Next, each client initiates a Neural Network with the received weights. Each client will iteratively train the network and send the local weights to the server, the server will average all the local weights and send back the results. Each client will set the received weights on their local Neural Network. This process is executed until convergence, when the average of all local weights is equal to the one in the previous iteration or when a maximum number of iterations is reached. With the test set, each user tests the model performance and sends the results to the server.

In order to not overload the validation phase with too many hyper-parameters, we designed a simple test where we created Neural Networks with a hidden layer of sizes 50, 100, 150, 200, 250, 300, and 500 and tested in the fixed scenario, where we used hierarchical clustering with 3 clusters. The results showed that a hidden layer of size 100 achieved the best results, although the changes in performance observed with other configurations were not significant. Therefore, the results presented in the following sections were achieved with a neural network with 1 hidden layer with 100 neurons and a single neuron output layer.

5 EVALUATION

In this section we assess the feasibility and performance of building profiles using the privacy-preserving strategies described in section 3.2. Since there is no ground truth for the generated user privacy profiles (see, for example, figure 4), this makes it harder to evaluate their usefulness in automating privacy decisions. To address this issue, in order to evaluate the utility of the generated profiles, we resort to the grant prediction results, that is, the performance of the neural network using privacy profiles after training with federating learning as aforementioned.

Using the strategies described before, we can use the grant prediction evaluation metrics to compare the usefulness of the different profiles generated. If the neural network is capable of achieving a better score using a specific set of profiles, we consider them to be more useful than another set of profiles, that achieves a lower score.

We use three metrics to evaluate the model's performance: F1-Score, Accuracy, and Precision-Recall AUC (Area Under the Curve). The F1-Score is the harmonic mean of the precision and recall, and we use it because it takes both false positives and false negatives into account, and this is also the metric used for comparison with previous works. Given the uneven class distribution in our dataset, the F1-Score is not always possible to calculate, *e.g.* when the dataset only contains one class. Therefore, we additionally consider the accuracy to give us an idea of how the model is behaving in these cases. Finally, the Precision-Recall AUC (PR-AUC) summarizes the Precision-Recall curve, and can be used to understand the trade-off in performance for different threshold values when interpreting probabilistic predictions. For the evaluation of the performance, we divided the dataset in 80% for the validation and 20% for testing, as described in section 5.1 and section 5.2, respectively.

5.1 Validation

To find the best set of profiles in our dataset we performed a grid search on the following parameters:

- Clustering Algorithm.
- Number of Clusters.

The clustering algorithms consist of the centralized (**hc centralized**) and the privacy preserving distributed (**distributed hc**) hierarchical clustering from sections 3.2.1 and 3.2.2, respectively, and the centralized (**centralized *k*-means**) and efficient privacy preserving distributed *k*-means (**distributed *k*-means**) from sections 3.2.3 and 3.2.4, respectively. The centralized algorithms that we used were just to understand if the distributed clustering algorithms were undermining the strategy's performance. As such, if we want to apply this to a real scenario we need to only conside the performance of the distributed clustering algorithms. The imputation method used was the multivariate imputer, that estimates each feature from all the others. But applied in different ways, one globally, using all the users data together (not possible in a private distributed manner) and applied locally, each user applies the imputation locally to their dataset. For every combination we applied

	Federated Learning					
	Accuracy		F1-Score		PR-AUC	
Best	0.88	($k = 9$)	0.91	($k = 9$)	0.98	($k = 10$)
	Distributed	k-Means	Distributed	k-Means	Distributed	HC
Worst	0.82	($k = 4$)	0.87	($k = 3$)	0.93	($k = 3$)
	Distributed	HC	Distributed	HC	Distributed	HC

Table 4: Best and worst results for federated learning to predict the grant result with the distributed hierarchical and k-means clustering approaches for generating privacy profiles.

a 5-fold cross validation, using 80% of the dataset, leaving 20% for testing.

Figure 1 presents the F1-Score for each of the clustering strategies as a function of the number of profiles. From this plot, we can see that the lowest value, for the distributed clustering strategies, is 0.87 with distributed hierarchical clustering using 3 profiles, and the highest one is 0.91 with distributed k-means with 9 profiles. It is also observable that the k-means algorithm, both centralized and distributed, outperform most of the strategies. Finally, this plot evidences the similarity in performance between the centralized and distributed approaches.

Table 4 presents the best and worst obtained performances for the distributed techniques and the three metrics: accuracy, F1-score and the precision-recall AUC. From the displayed results we can conclude that the lowest accuracy is 0.82 with distributed hierarchical clustering with 4 profiles, and the highest is 0.88 with distributed k-means with 9 profiles. In fact, the distributed hierarchical clustering sees the lowest results in the three metrics. Finally, the precision-recall AUC scores present a more optimistic view, with the lowest score being 0.93 for distributed hierarchical clustering using 3 profiles, and the highest being 0.98 for distributed hierarchical clustering using 10 profiles.

Overall, the best secure model is the distributed k-means using 9 profiles (c.f. table 4), with an F1-score of 0.88. This demonstrates that our strategy can be used in a fully distributed scenario, where both the profiles' generation and evaluation are done in a private distributed manner.

5.2 Testing

In order to test the best models found in the validation phase, each user will train the models using the data from validation, 80% of their dataset, and test it with the remaining 20%. The F1-score, accuracy and precision-recall AUC metrics for each user is presented in figure 2 as a scatter plot as a function of the percentage of granted permission fitted through a linear regression as to identify any bias in the model, that is, to identify potential skewness originating from users that always allow or always deny requests. Despite a lower performance for users that deny most requests (i.e. lower percentage of granted permissions), this plot evidences overall good performances. In fact, the global F1-score was 0.90, the global accuracy was 0.88 and the global precision-recall AUC was 0.97. These results are positive and comparable to the centralized results obtained by Liu, Lin and Sadeh [12], where the prediction of the grant result with a linear-kernel support vector machine using hierarchical clustering to generate the privacy profiles achieved a cross-validated F1-Score of 0.9002. In our federated/distributed

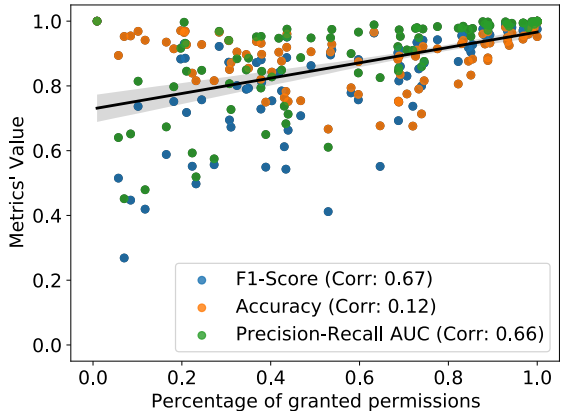


Figure 2: Scores as a function of the percentage of granted permissions for each user (each user is one point) in the federated learning. For each metric we also provide the Person's correlation between the two variables (Corr).

approach, we are able to achieve a performance comparable to the centralized approach with an F1-score of 0.90 (0.744 without profiles), with the privacy advantage that our mechanisms allow for the creation of user profiles and the training of neural network prediction models to be done locally in a privacy preserving manner with minimal reliance on a central server.

6 FUTURE WORK

In the future, to improve the robustness and applicability of our strategy, data from broader demographics should be collected. This data would allow us to analyse more diverse responses to different app categories and permissions, thus testing the robustness of our strategy. It would also be interesting to perform a noise analysis on the dataset, as well as a more extensive test on the adequate number of privacy profiles, in order to find the saturation point.

We also need to build a framework capable of monitoring the model's performance in a secure way, together with the ability to update the profiles with the new data in a secure way as well. Such framework would improve the adaptability of the model to real world scenarios where variability of choices may occur, and thus incorporate changes in privacy preferences throughout time.

The final strategy we presented to predict the users' grant decisions is complex, including two learning phases: one for privacy profile generation, another for using the profiles and context variables to predict the users' answers. For future work, a deep learning

approach together with federated learning could be capable of replacing the two step process by a single one.

7 CONCLUSION

In this paper we present methods for generating privacy profiles and using these to predict user's answers to permission requests in mobile devices. The prediction and generation of privacy profiles is performed with privacy guarantees, not requiring access to user data, unlike the state of the art in the area. Towards this end we resort to privacy preserving clustering techniques to generate the profiles, while maintain the client's privacy, even against the server computing the profiles. By combining the process to generate the profiles privately with federated learning techniques, we were able to demonstrate the usefulness of the privacy profiles in predicting users' grant decisions in a secure fashion, *i.e.* without sharing user data. Our strategy was evaluated with a dataset of 93 participants obtained from a field-study, thus showing that the proposed techniques can be applied in real-world scenarios to generate privacy profiles and predict users' permission requests while preserving user privacy. Moreover, our secure and distributed strategy achieved an F1-score of 90%, matching the centralized state-of-the-art performance for prediction of permission requests.

ACKNOWLEDGEMENTS

This work is supported by project COP-MODE, that has received funding from the European Union's Horizon 2020 research and innovation programme under the NGI TRUST grant agreement no 825618, and the project SNOB-5G with Nr. 045929 (CENTRO-01-0247-FEDER-045929) supported by the European Regional Development Fund (FEDER), through the Regional Operational Programme of Centre (CENTRO 2020) of the Portugal 2020 framework and FCT under the MIT Portugal Program. Ricardo Mendes wishes to acknowledge the Portuguese funding institution FCT - Foundation for Science and Technology for supporting his research under the Ph.D. grant SFRH/BD/128599/2017.

REFERENCES

[1] Yuvraj Agarwal and Malcolm Hall. 2013. ProtectMyPrivacy: Detecting and Mitigating Privacy Leaks on IOS Devices Using Crowdsourcing. In *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services* (Taipei, Taiwan) *(MobiSys '13)*. Association for Computing Machinery, New York, NY, USA, 97–110. https://doi.org/10.1145/2462456.2464460

[2] Panagiotis Andriotis, Gianluca Stringhini, and Martina Angela Sasse. 2018. Studying users' adaptation to Android's run-time fine-grained access control system. *Journal of Information Security and Applications* 40 (2018), 31–43. https://doi.org/10.1016/j.jisa.2018.02.004

[3] André Brandão, Ricardo Mendes, and João P Vilela. 2021. Efficient privacy preserving distributed K-means for non-IID data. In *Advances in Intelligent Data Analysis XIX*. Springer International Publishing, Cham, 439–451.

[4] Wenyun Dai, Meikang Qiu, Longfei Qiu, Longbin Chen, and Ana Wu. 2017. Who Moved My Data? Privacy Protection in Smartphones. *IEEE Communications Magazine* 55 (01 2017), 20–25. https://doi.org/10.1109/MCOM.2017.1600349CM

[5] Adrienne Porter Felt, Elizabeth Ha, Serge Egelman, Ariel Haney, Erika Chin, and David Wagner. 2012. Android Permissions: User Attention, Comprehension, and Behavior. In *Proceedings of the Eighth Symposium on Usable Privacy and Security* (Washington, D.C.) *(SOUPS '12)*. Association for Computing Machinery, New York, NY, USA, Article 3, 14 pages. https://doi.org/10.1145/2335356.2335360

[6] Alessandra Gorla, Ilaria Tavecchia, Florian Gross, and Andreas Zeller. 2014. Checking App Behavior against App Descriptions. In *International Conference on Software Engineering* (Hyderabad, India). Association for Computing Machinery, New York, NY, USA, 1025–1035. https://doi.org/10.1145/2568225.2568276

[7] Mona Hamidi, Mina Sheikhalishahi, and Fabio Martinelli. 2018. A Secure Distributed Framework for Agglomerative Hierarchical Clustering Construction. In *2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing*. IEEE, UK, 430–435. https://doi.org/10.1109/PDP2018.2018.00075

[8] International Data Corporation. 2018. Smartphone Market Share. https://www.theguardian.com/news/2018/mar/17/cambridge-analytica-facebook-influence-us-election. Accessed: 2021-10-11.

[9] Qatrunnada Ismail, Tousif Ahmed, Apu Kapadia, and Michael K. Reiter. 2015. *Crowdsourced Exploration of Security Configurations*. Association for Computing Machinery, New York, NY, USA, 467–476. https://doi.org/10.1145/2702123.2702370

[10] Patrick Gage Kelley, Sunny Consolvo, Lorrie Faith Cranor, Jaeyeon Jung, Norman Sadeh, and David Wetherall. 2012. A Conundrum of Permissions: Installing Applications on an Android Smartphone. In *Financial Cryptography and Data Security*, Jim Blyth, Sven Dietrich, and L. Jean Camp (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 68–79.

[11] Jialiu Lin, Bin Liu, Norman Sadeh, and Jason I. Hong. 2014. Modeling Users' Mobile App Privacy Preferences: Restoring Usability in a Sea of Permission Settings. In *Proceedings of the Tenth USENIX Conference on Usable Privacy and Security* (Menlo Park, CA) *(SOUPS '14)*. USENIX Association, USA, 199–212.

[12] Bin Liu, Mads Schaarup Andersen, Florian Schaub, Hazim Almuhimedi, Shikun Zhang, Norman Sadeh, Alessandro Acquisti, and Yuvraj Agarwal. 2016. Follow My Recommendations: A Personalized Privacy Assistant for Mobile App Permissions. In *Proceedings of the Twelfth USENIX Conference on Usable Privacy and Security* (Denver, CO, USA) *(SOUPS '16)*. USENIX Association, USA, 27–41.

[13] Bin Liu, Jialiu Lin, and Norman Sadeh. 2014. Reconciling Mobile App Privacy and Usability on Smartphones: Could User Privacy Profiles Help?. In *Proceedings of the 23rd International Conference on World Wide Web* (Seoul, Korea) *(WWW '14)*. Association for Computing Machinery, New York, NY, USA, 201–212. https://doi.org/10.1145/2566486.2568035

[14] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 54)*. PMLR, 1273–1282.

[15] Ricardo Mendes, André Brandão, J. P. Vilela, and Alastair R. Beresford. 2022. Effect of User Expectancy on Mobile App Privacy: A Field Study. In *2022 IEEE international conference on pervasive computing and communications (PerCom)*.

[16] Patricia A. Norberg, Daniel R. Horne, and Dadid A. Horne. 2007. The Privacy Paradox: Personal Information Disclosure Intentions versus Behaviors. *The Journal of Consumer Affairs* 41, 1 (2007), 100–126.

[17] Katarzyna Olejnik, Italo Dacosta, Joana Soares Machado, Kévin Huguenin, Mohammad Emtiyaz Khan, and Jean-Pierre Hubaux. 2017. Smarper: Context-aware and automatic runtime-permissions for mobile devices. In *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1058–1076.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[19] Bahman Rashidi, Carol Fung, and Tam Vu. 2015. Dude, ask the experts!: Android resource access permission recommendation with RecDroid. In *2015 IFIP/IEEE International Symposium on Integrated Network Management*. IEEE, UK, 296–304.

[20] Ramprasad Ravichandran, Michael Benisch, Patrick Gauge Kelley, and Norman Sadeh. 2009. Capturing Social Networking Privacy Preferences: Can Default Policies Help Alleviate Tradeoffs between Expressiveness and User Burden?. In *Proceedings of the 5th Symposium on Usable Privacy and Security* (Mountain View, California, USA) *(SOUPS '09)*. Association for Computing Machinery, New York, NY, USA, Article 47, 1 pages. https://doi.org/10.1145/1572532.1572587

[21] Lena Reinfelder, Andrea Schankin, Sophie Russ, and Zinaida Benenson. 2018. An Inquiry into Perception and Usage of Smartphone Permission Models. In *Trust, Privacy and Security in Digital Business*, Steven Furnell, Haralambos Mouratidis, and Günther Pernul (Eds.). Springer International Publishing, Cham, 9–22.

[22] Odnan Ref Sanchez, Ilaria Torre, Yangyang He, and Bart Knijnenburg. 2020. A recommendation approach for user privacy preferences in the fitness domain. *User Modeling and User-Adapted Interaction* 30 (07 2020). https://doi.org/10.1007/s11257-019-09246-3

[23] United Nation General Assembly. 1948. Universal Declaration of Human Rights. , 6 pages. https://doi.org/10.1080/13642989808406748 arXiv:arXiv:1011.1669v3

[24] Jaideep Vaidya and Chris Clifton. 2002. Privacy Preserving Association Rule Mining in Vertically Partitioned Data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Edmonton, Alberta, Canada) *(KDD '02)*. Association for Computing Machinery, New York, NY, USA, 639–644. https://doi.org/10.1145/775047.775142

[25] Xuetao Wei, Lorenzo Gomez, Iulian Neamtiu, and Michalis Faloutsos. 2012. Permission Evolution in the Android Ecosystem. In *Proceedings of the 28th Annual Computer Security Applications Conference* (Orlando, Florida, USA) *(ACSAC '12)*. Association for Computing Machinery, New York, NY, USA, 31–40. https://doi.org/10.1145/2420950.2420956

[26] Jierui Xie, Bart Piet Knijnenburg, and Hongxia Jin. 2014. Location Sharing Privacy Preference: Analysis and Personalized Recommendation. In *Proceedings*

of the 19th International Conference on Intelligent User Interfaces (Haifa, Israel) *(IUI '14)*. Association for Computing Machinery, New York, NY, USA, 189–198. https://doi.org/10.1145/2557500.2557504

[27] Yuchen Zhao, Juan Ye, and Tristan Henderson. 2014. Privacy-Aware Location Privacy Preference Recommendations. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services* (London, United Kingdom) *(MOBIQUITOUS '14)*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), Brussels, BEL, 120–129. https://doi.org/10.4108/icst.mobiquitous.2014.258017

A DATA CHARACTERIZATION

In this appendix we present figure 3 that represents the average grant result answered from each user according to the app category and permission.

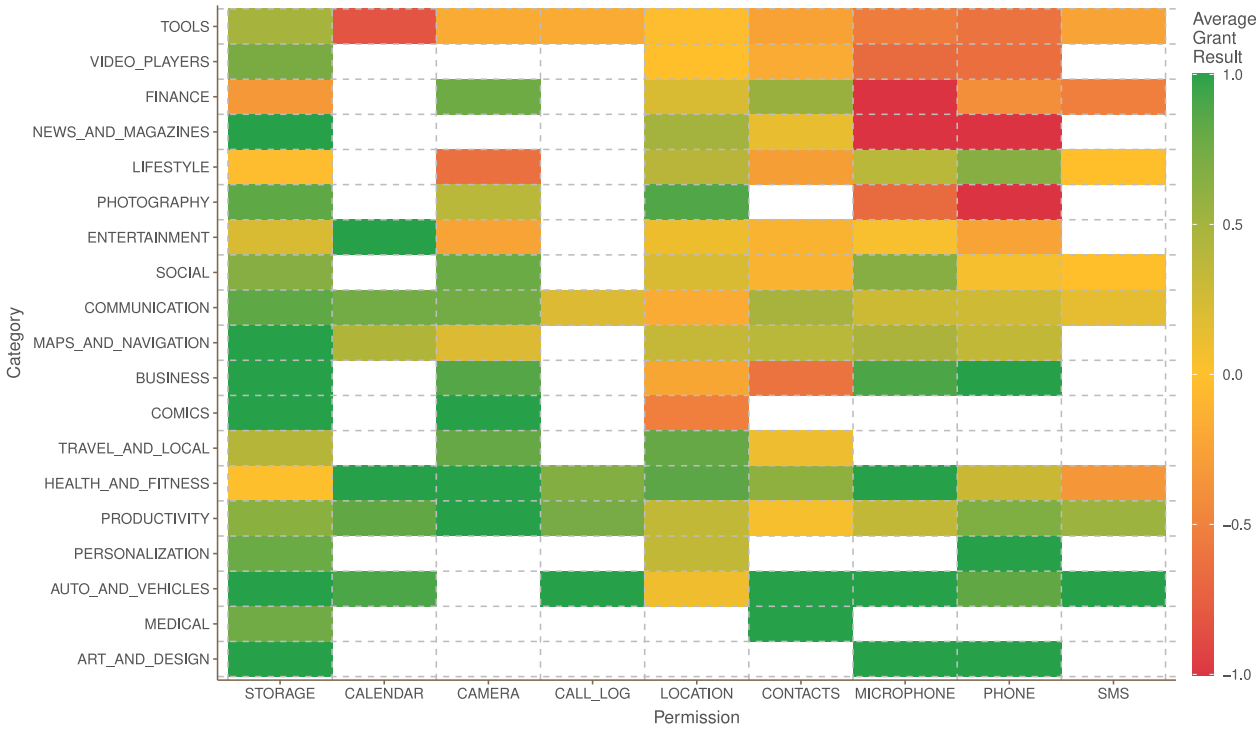
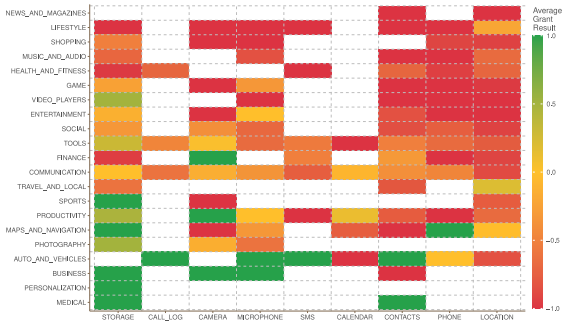


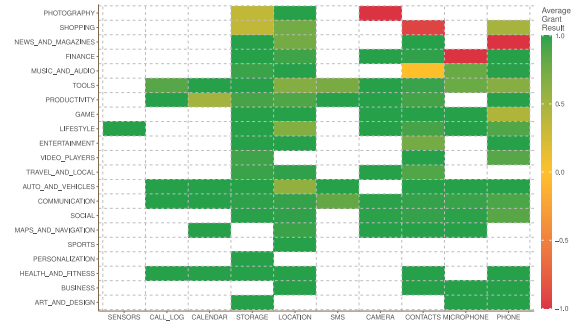
Figure 3: Average grant result decision for app category and permission.

B PRIVACY PROFILES VISUALISATION

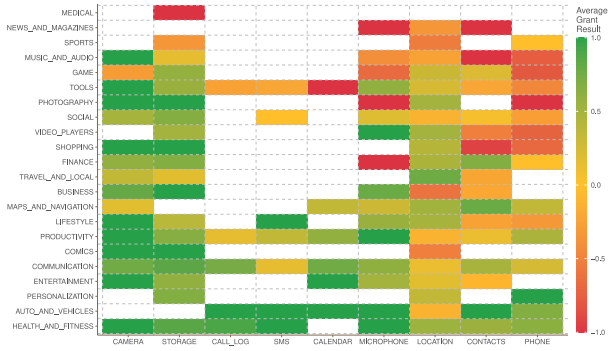
In this appendix we present figure 4, containing three example profiles, resulting from hierarchical clustering.



(a) Profile 1 - the privacy conscious user.



(b) Profile 2 - permissive user.



(c) Profile 3 - the middle ground user.

Figure 4: Privacy profiles. (a) Profile of the privacy conscious. (b) Profile of the permissive users. (c) Profile of the "middle-ground" users.

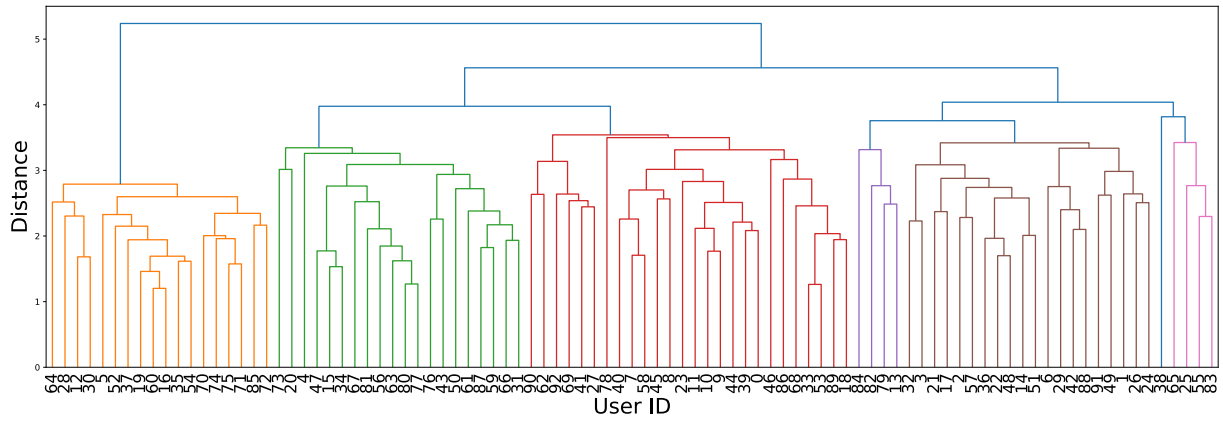


Figure 5: Resulting dendogram.

C FEDERATED LEARNING TRAINING DIAGRAM

In this appendix we present figure 6, where a thorough description of the federated learning training algorithm.

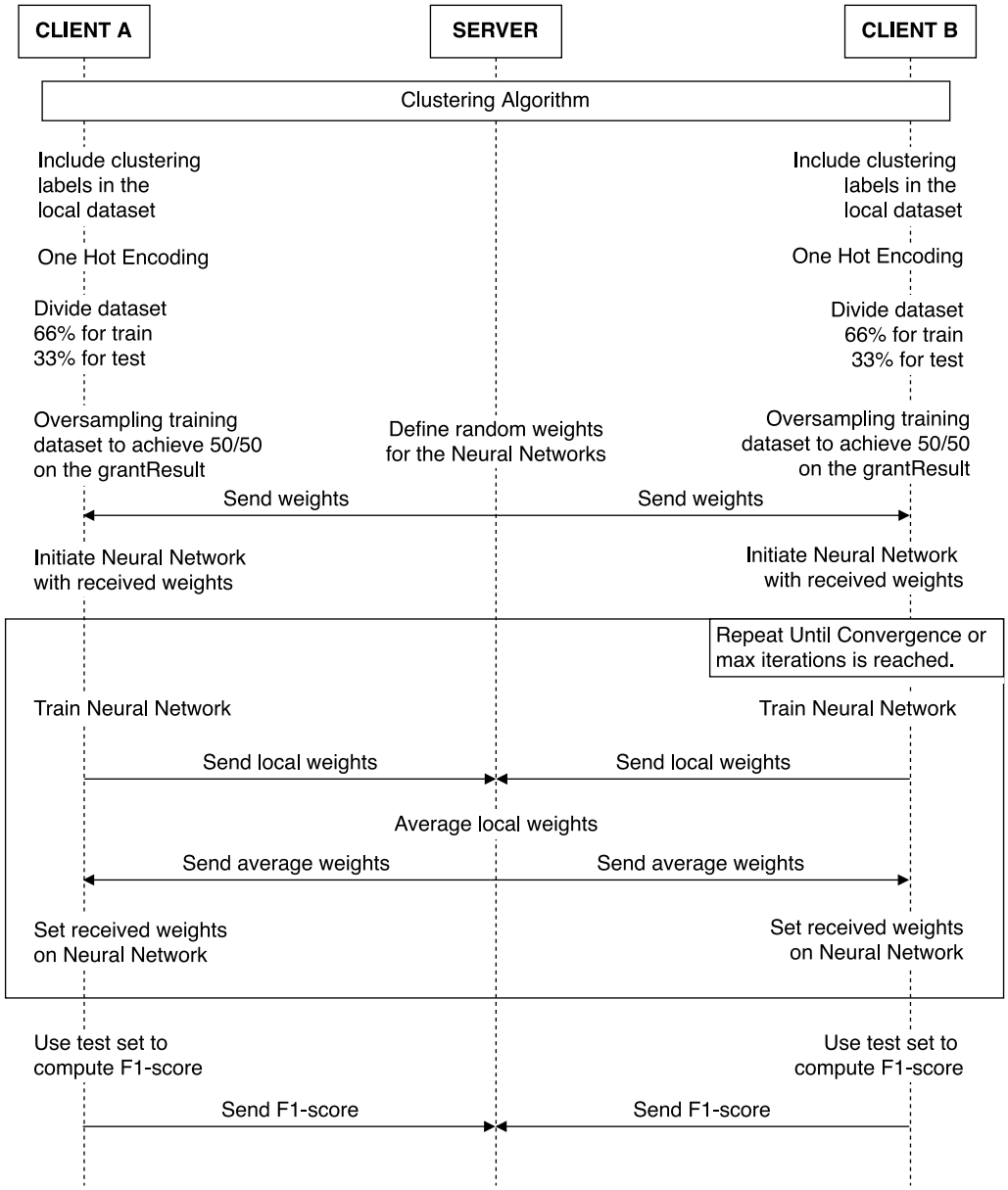


Figure 6: Federated learning training diagram.