MOBIDATALAB

Labs for prototyping future mobility data sharing solutions in the cloud

D4.9 Data Protection Tools V1

The anonymization module of the MobiDataLab transport cloud prototype.

23/01/2023

Author(s): Jesús A. MANJON (URV), Alberto BLANCO (URV), Sergio MARTINEZ (URV), Benet MANZARES (URV)

MobiDataLab is funded by the EU under the H2020 Research and Innovation Programme (grant agreement No 101006879).

Summary sheet

Deliverable Number	D4.9
Deliverable Name	Data Protection Tools V1
Full Project Title	MobiDataLab, Labs for prototyping future Mobility Data sharing cloud solutions
Responsible Author(s)	Jesús A. MANJON (URV)
Contributing Partner(s)	-
Peer Review	HOVE, HERE
Contractual Delivery Date	31-07-2022
Actual Delivery Date	28-07-2022
Status	Final
Dissemination level	Public
Version	V1.0
No. of Pages	28
WP/Task related to the deliverable	WP4/T4.5
WP/Task responsible	AKKA/URV
Document ID	MobiDataLab-D4.9-DataProtectionToolsV1-v1.0
Abstract	This document presents the anonymization module of the MobiDataLab transport cloud prototype. The anonymization tool includes methods for the protection of mobility data and the computation of privacy and utility metrics.

Legal Disclaimer

MOBIDATALAB (Grant Agreement No 101006879) is a Research and Innovation Actions project funded by the EU Framework Programme for Research and Innovation Horizon 2020. This document contains information on MOBIDATALAB core activities, findings, and outcomes. The content of this publication is the sole responsibility of the MOBIDATALAB consortium and cannot be considered to reflect the views of the European Commission.

Project partners

Organisation	Country	Abbreviation
Universitat Rovira i Virgili	Spain	URV
Consiglio Nazionale delle Ricerche	Italy	CNR
AKKA I&S	France	AKKA

Document history

Version	Date	Organisation	Main area of changes	Comments
0.1	16/06/2022	URV	Table of contents	TOC
0.2	07/07/2022	URV	All	Draft
0.3	13/07/2022	HERE, HOVE	All	Peer review
0.4	27/07/2022	URV	All	Rework
0.5	27-28/07/2022	AKKA	All	Quality check
1.0	28/07/2022	AKKA	All	Final version

Funded by the European Union

Executive Summary

The main goal of task T4.5 is to develop data processing modules that apply data protection and anonymization techniques to the data in the Transport Cloud.

The demonstrator currently includes 3 anonymization methods for protecting trajectory data, selected from the catalogue of techniques compiled in T2.2 considering the use case requirements elicited in T2.6, and the computation of utility metrics in trajectory databases. It has been designed with a focus on modularity, where pseudonymization or anonymization methods can be built using different components dedicated to pre-processing, clustering, distance computation, aggregation, etc. The developed package also provides a command line interface (CLI) that lets users anonymize a mobility dataset and compute some utility measures over both the original and the anonymized datasets in a straightforward way.

Deliverable D4.9 describes the characteristic of the demonstrator and includes a detailed user manual. Demonstrator is available at https://github.com/MobiDataLab/mdl-anonymizer.

Table of contents

List of Figures

List of Tables

Abbreviations and acronyms

Abbreviation	Meaning
MDAV	Maximum Distance to Average Vector
CLI	Command Line Interface
DTW	Dynamic time warping
STLC	Spatio-temporal linear combine

Funded by the European Union

Introduction

1.1. Project overview

1.

There has been an explosion of mobility services and data sharing in recent years. Building on this, the EU-funded MobiDataLab project works to foster the sharing of data amongst transport authorities, operators, and other mobility stakeholders in Europe. MobiDataLab develops knowledge as well as a cloud solution aimed at easing the sharing of data. Specifically, the project is based on a continuous co-development of knowledge and technical solutions. It collects and analyses the advice and recommendations of experts and supporting cities, regions, clusters, and associations. These actions are assisted by the incremental construction of a cross-thematic knowledge base and a cloud-based service platform, which will improve access and usage of data-sharing resources.

1.2. Purpose of this deliverable

This document presents the anonymization module of the MobiDataLab Transport Cloud prototype. The anonymization tool includes methods for the protection of mobility data and the computation of privacy and utility metrics. This version of the module is limited to the protection of trajectory data and the computation of utility metrics in trajectory databases. The currently implemented methods are described in detail in Section 6 of D2.3 - State of the Art on Transport and Mobility Data Protection Technologies. A user manual is provided in Section 4. This document is a companion report to the demonstrator available at https://github.com/MobiDataLab/mdl-anonymizer.

1.3. Structure of the deliverable

This deliverable is organized as follows. Section 2 describes the design of the anonymization module, including class and use case diagrams. Section 3 documents the components implemented in the current version of the module. Section 4 is the user manual for the CLI implementation of the tool. Finally, Section 5 presents the conclusions and a roadmap for additional features to be implemented for the next version of the tool.

Funded by the European Union

2. Design

In D2.3 State of the Art on Transport and Mobility Data Protection Technologies, we studied a collection of anonymization methods for trajectory microdata in the literature. This study suggested that methods in that category tend to follow a common pattern that consists of data pre-processing, comparison or clustering of data according to some spatial (or spatiotemporal) distance metric between positions and/or trajectories and an optional final aggregation or filtering step. For this reason, we design our anonymization tool with a focus on modularity, where pseudonymization or anonymization methods can be built using different components dedicated to pre-processing, clustering, distance computation, aggregation, etc.

On the other hand, we want our module to be as format-agnostic as possible, and thus we choose to load data from comma-separated values (CSV), which makes it directly compatible with tools such as QGIS and mobility data analysis libraries such as GeoPandas and scikit-mobility. This decision, however, does not limit the possibility of adding additional data loading components to deal with different data formats.

2.1. Entity class diagram

Entity classes implement the data model and the non-interactive functionalities (*i.e.*, anonymization mechanisms) of the anonymization module. To achieve high performance, classes have been designed to be cohesive and decoupled, and associations between them have been defined with the logical navigation workflow. The main highlights of the class diagram, which is depicted in Figure 1, are:

- *Dataset* represents a data set of trajectories. Datasets are structured in several classes interrelated with navigable associations (*Dataset* → *Trajectory* → *TimestampedLocation*) so that all the data associated with a data set (*i.e.,* trajectory values) can be efficiently queried during the anonymization process.
- Anonymization algorithms are defined as specializations of the *AnonymizationScheme* abstract class, which defines the interfaces of the main operations (anonymization) and implements the common trajectory management operations (computation of utility and privacy metrics). Thus, new anonymization algorithms or variations of the three currently implemented ones can be easily added by specializing classes and reusing or extending the code.
- Distance computation algorithms for trajectories are defined as specializations of the *DistanceInterface* interface, which defines the operation of computing the distance between two trajectories. Thus, new distance computation algorithms or variations of the two currently implemented ones can be easily added by implementing the distance interface.

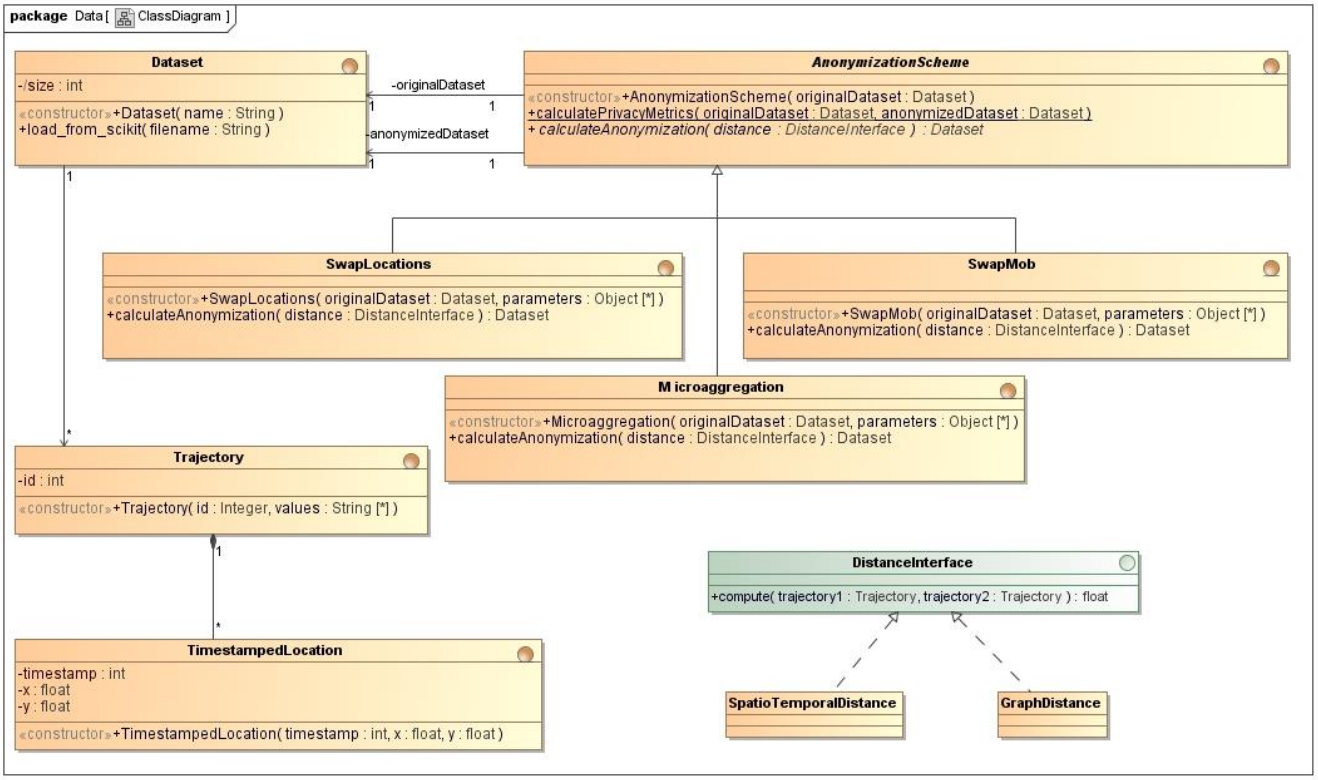

Figure 1 - Entity class diagram

2.2. Identification of use cases

Figure 2 shows the use cases diagram of the processor based on the functional requirements. We consider only 1 actor, named user, who directly interacts with the software. We identify the following use cases:

1. *Create a configuration file*, which creates and provides a JSON configuration file to the module which includes the parameter values to configure the anonymization methods (see section 4.1). The JSON configuration file must be manually created.
2. *Anonymize the dataset*, which, given a configuration file and a data set of trajectories, produces an anonymized version of the data set.
3. *Calculate utility and privacy metrics*, which, given original and anonymized trajectory data sets, computes and displays the utility and privacy metrics of the anonymized data set with respect to the original data set.

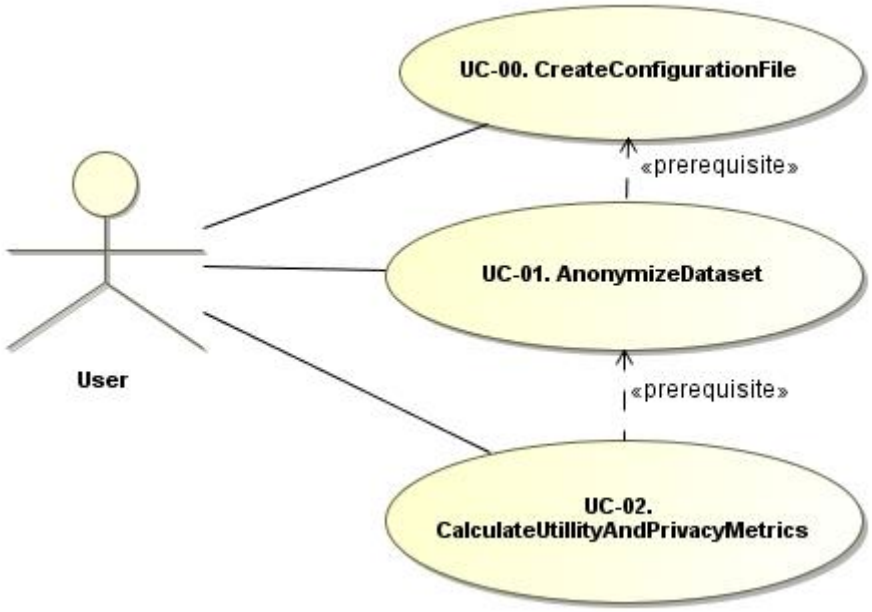

Figure 2 - Use cases diagram

Funded by the European Union

3. Developed methods

This section describes the software components implemented in the current version of the anonymization module. As described above, anonymization mechanisms can be built from different components, including distance computation between locations and trajectories, aggregation and clustering algorithms, and postprocessing operations.

3.1. Trajectory distances

Many of the trajectory anonymization mechanisms in the literature (refer to D2.3, Section 6) consist, in some way, in reducing the unicity of trajectories typically by grouping or aggregating similar ones. This similarity is measured with a distance metric to quantify the resemblance of the trajectories to be grouped or aggregated. This distance must consider both the space and time dimensions. In the following subsections, we describe the methods to compute a distance between two trajectories that have been developed so far and integrated into the anonymization module.

3.1.1. Graph distance

This method, presented in [1], is based on the computation of a spatial distance between pairs of trajectories only when they are 'contemporaries' (that is, they overlap in time). The spatial distance between each pair of trajectories is only computed for locations within the time interval that they share. Then a graph is built where i) the nodes represent trajectories; ii) nodes T_i and T_j are adjacent only if they are contemporaries, and iii) the weight of the edge (T_i, T_j) is the distance between the trajectories T_i and T_j. Given the distance graph for $T = \{T_1, \ldots, T_n\}$, the distance $d(T_i, T_j)$ for two trajectories is easily computed as the minimum cost path between the nodes T_i and T_j, if such a path exists.

Figure 3 shows an example of a distance graph. T_1 and T_2 are adjacent because they overlap in time and a distance exists between them. However, T_2 and T_4 are not connected because they are not contemporaries, and their distance is not defined. T_1 and T_8 are not connected (they do not overlap in time) but their distance can be computed as the minimum cost path between the nodes, in this case, d = 6.01.

Funded by the European Union

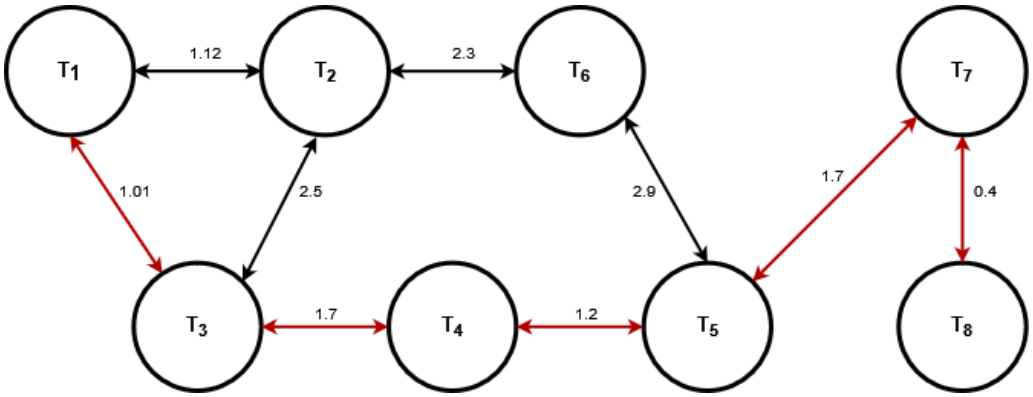

Figure 3 - A distance graph. Red arrows are the shortest path between T_1 and T_8.

3.1.2. Spatio-temporal distance

Most of the spatial distances can be extended into Spatio-temporal distances by balancing the weight of the spatial and temporal dimensions [6]. Additionally, since each trajectory may have a different number of points, a method to sample, match and compare individual points within trajectories is needed.

Dynamic time warping (DTW) [3] is one of the most popular algorithms to measure the distance between trajectories. The DTW algorithm searches through all locations in two trajectories for a pair of points at a minimum distance. The computational cost of DTW is $O(mn)$, where *m* and *n* are the numbers of points in each of the trajectories. In [5], the authors propose the Spatio-temporal linear combine (STLC) distance, where spatial and temporal similarities are linearly combined according to a parameter λ which assigns a weight to both time and space similarities. The computational cost of STLC is quadratic $O(mn)$.

In [2] we defined a distance measure based on DTW and STLC that tries to find the best match between pairs of points in the trajectories with a low computation cost. The computational cost of the distance calculation is just O(*h*) where *h* is the average number of points in the two trajectories, which makes this distance suitable for large data sets. To calculate the distance, the algorithm selects a list of *h* representative pairs of points, proportional to the number of points in each trajectory. Only these points are considered during the distance calculation. Figure 4 shows an example of the pairs of points selected to compare trajectories T_a and T_b. Origin and destination points of the trajectories are always selected.

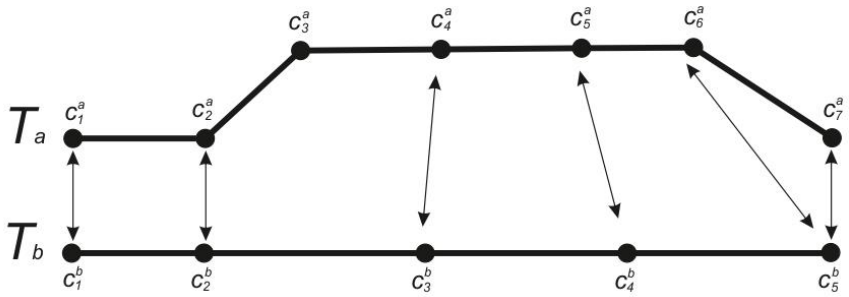

Figure 4 - Trajectory distance calculation. Arrows show the pairs of points selected to calculate the distance between trajectories T_a and T_b

Once a pair of points have been matched, the distance between them is computed. Similar to STLC, we define a distance that linearly combines spatial and temporal distances between pairs of points:

$$dist(c_i, c_j) = dist_{spa}\left((x_i^a, y_i^a), (x_j^b, y_j^b)\right) + \lambda \cdot (|t_i^a - t_j^b| \cdot V_{ab})$$

where $dist_{spa}$ is the spatial distance between coordinates (x_i^a, y_i^a) and (x_j^b, y_j^b), and V_{ab} is the mean velocity of trajectories T_a and T_b. The temporal distance $|t_i^a - t_j^b|$ is multiplied by the mean velocity of trajectories V_{ab} to convert it to a spatial distance that can be added to $dist_{spa}$. To mitigate the excessive weight of the temporal distance (because implicitly assumes that subjects are constantly moving away from each other) we weight the temporal component with a parameter $\lambda = \frac{D}{V \cdot T}$, where *D* is the maximum distance between points in the data set, *V* is the mean velocity of the trajectories in the data set and *T* is the maximum time difference between points in the data set.

3.2. Trajectory aggregation

Trajectory aggregation consists in replacing a set of trajectories with a single representative, namely the centroid of the set. Since the replacement causes information loss, the calculation of accurate centroid trajectories is crucial to retain the utility of the anonymized data set as much as possible.

3.2.1. Mean trajectory

In [2], we propose a trajectory aggregation algorithm with a linear computational cost which makes it suitable for large data sets. The algorithm works as follows: being Q a set of trajectories to be aggregated, the trajectory aggregation algorithm calculates h as the mean of points in the trajectories included in Q. Similarly, to how the distance computation presented in **Error! Reference source not found.** works, we select a sample of *h* points from each trajectory in Q, proportionally to the number of points. This yields *h* sets of $|Q|$ points each. The centroid (t^c, x^c, y^c) of each of the h sets is calculated as the component-wise mean of the $|Q|$ points in the set. Finally, the centroid trajectory Q_c is obtained as the concatenation of the *h* centroids, that is, $Q_c = \{(t_1^c, x_1^c, y_1^c), \ldots, (t_h^c, x_h^c, y_h^c)\}$.

Figure 5 shows an example of the aggregation of two trajectories T_a and T_b. The arrows show the sample points selected from each trajectory. The resulting centroid trajectory Q_c is depicted with a dotted line.

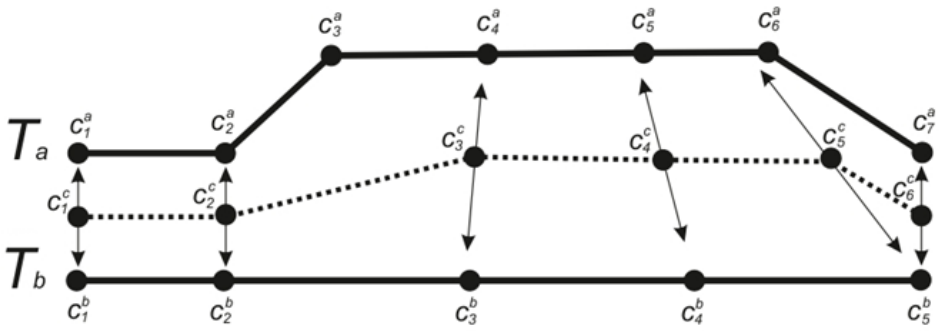

Figure 5 - Aggregation of trajectories. The arrows show the points selected to aggregate the trajectories T_a and T_b. The dotted line represents the centroid trajectory taken as the output of the aggregation

3.3. Anonymization methods

This section presents the anonymization mechanisms that have been implemented in the anonymization module up to this point. This list of methods is expected to be extended in a future version of the module.

3.3.1. Swap locations

This method is based on the *ReachLocation* mechanism proposed by [1]. Protection is achieved by permuting the locations in trajectories among other trajectories. The method works as follows: first, a cluster is created around a randomly selected location based on some spatial and temporal threshold parameters provided by the user. If the cluster does not include at least k locations from at least k different trajectories, the thresholds are increased until we obtain the required number of locations, or the thresholds reach user-defined maximum values. If a valid cluster is built, the locations are swapped among the k trajectories (by changing their trajectory IDs) and marked as *swapped*. If no valid cluster can be found around a location, it is removed. This process continues until no more “unswapped” locations appear in the data set.

This method provides great utility since locations in the resulting anonymized trajectories are true, fully accurate original locations. No fake, generalized, perturbed locations are given in the anonymized data set of trajectories. Besides that, the flow and directions of the original trajectories are well preserved. However, this mechanism does not offer a formal guarantee of privacy. If a whole trajectory is unique, the user could be identified.

Figure 6 shows an example of this anonymization mechanism. The yellow circles are the original locations. In green are depicted all the locations from the original dataset that meet the spatial and temporal requirements to be swapped with one of the original locations to be anonymized. Finally, in red we can see the locations that have been selected to build the anonymized trajectory.

Figure 6 - Example of anonymization with the Swap Locations method

3.3.2. SwapMob

SwapMob, proposed by J. Salas, D. Megías and V. Torra [4] is a perturbative anonymization method based on swapping segments of trajectories with other trajectories. When two locations in two different trajectories are close enough (when they cross each other), according to some threshold of proximity and time set by the user, the two remaining subtrajectories are swapped between the two original trajectories (that is, the ID of the previous locations of each trajectory are swapped). Changing pseudonyms (IDs) is equivalent to swapping the partial trajectories. If a trajectory does not cross any other one, and therefore, no subtrajectory is swapped with other trajectories, it is removed.

Hence, the relationship between data subjects and their data is obfuscated while keeping precise aggregated data, such as the number of users and their directions in any given zone at a specific time, the locations that have been visited by different anonymous users or the average length of trajectories.

Nevertheless, this comes at the cost of modifying the trajectories and losing individual trajectory mining utility.

In Figure 7, we can see an example of the anonymization method. The anonymized trajectory (dark red) is made up of subsegments of several original trajectories. The first subsegment comes from the green trajectory and the following subsegments come from the pink, orange, yellow, purple, and blue trajectories.

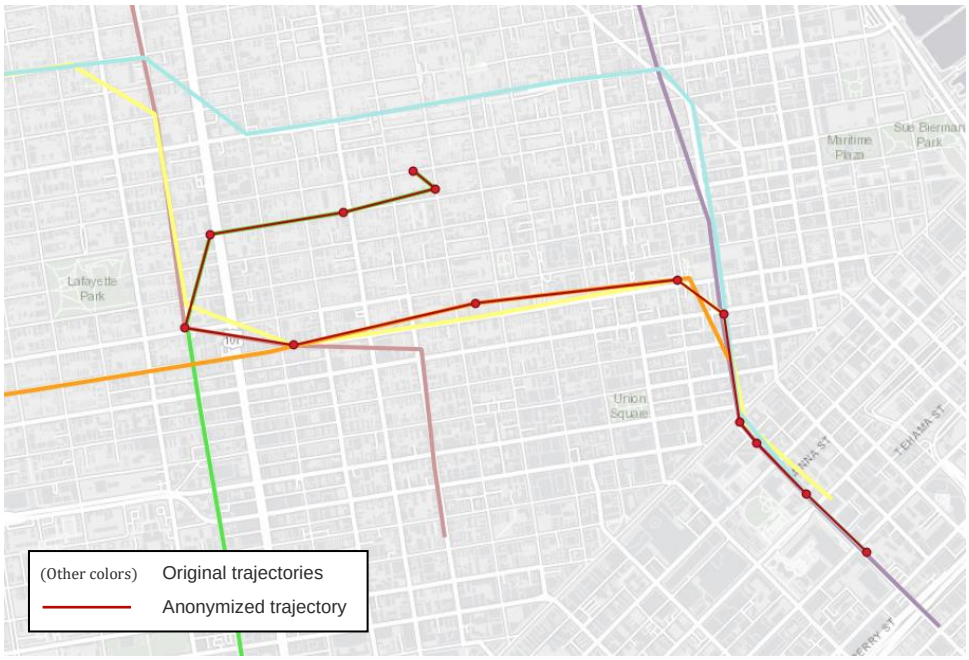

Figure 7 - Example of anonymization with the SwapMob method

3.3.3. Microaggregation

k-Anonymity limits the capability of an attacker who knows a set of features on a subject (some locations in the case of trajectory data) to perform successful re-identifications in a released data set. A trajectory data set satisfies *k*-anonymity if, for each combination of locations, at least k trajectories share the same combination. Thus, the probability of correct re-identification is, at most, $1/k$.

Even though *k*-anonymity has usually been enforced via generalization of values, this entails a large information loss for high-dimensional and spread data such trajectories. A more utility-preserving alternative to generalization is micro aggregation [2]. Trajectory microaggregation is based on partitioning the data set into disjoint clusters containing each at least *k* similar trajectories. Once the clustering of the data set is complete, the trajectories in each cluster are aggregated by replacing them with the cluster centroid. During the clustering stage, trajectories are grouped minimizing their intra-cluster distance (see section Trajectory distances3.1 trajectory distances). In the aggregation step, the centroid of the data set is calculated as the trajectory in the data set that minimizes the distance to the rest of the trajectories (see section 3.2 trajectory aggregation). In this manner, the resulting microaggregated data set minimizes the information loss incurred when enforcing *k*-anonymity.

Figure 8 shows an example of the anonymization of trajectories by microaggregation. The anonymized trajectory (red) is the result of the aggregation of *k* nearest trajectories (depicted in blue, green and orange) included in a cluster, in this case, of size $k=3$.

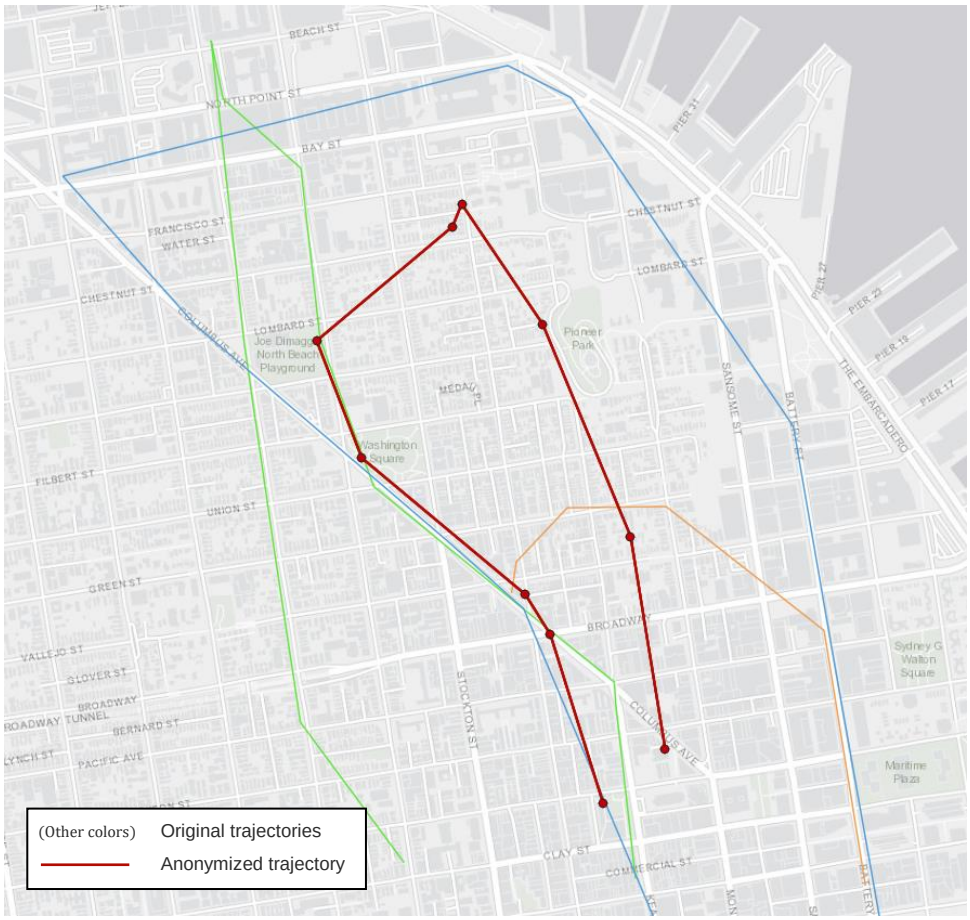

Figure 8 - Example of anonymization with the microagreggation method

3.3.4. Comparison of the anonymization methods

As described in the previous section, every anonymization method preserves different utility metrics. We applied all 3 implemented anonymization methods to the same dataset and summarize some utility results in Table 1.

The experiments were run on an Intel i5-8250U with 8 GB of RAM. We used a dataset built from the mobility data of taxi cabs in San Francisco, USA, provided by the Exploratorium Museum within the Cabspotting project[1]. The data set contains the trajectories of approximately 500 taxi cabs in the San Francisco Bay Area recorded during May 2008. The data capture the usual features of realistic trajectories (*i.e.*, short trajectories in areas with a dense population). Each record contains the GPS coordinates and absolute times of all trajectory points.

[1] http://www.exploratorium.edu/id/cab.html

We joined trajectories recorded between 07:00 and 07:15 throughout all the days in the data set into one single day to obtain a large and dense dataset. We considered only the trajectories that correspond to cabs that were occupied by a customer. This resulted in realistic trajectories with meaningful and precise origins, paths, and destinations, rather than seemingly random routes of cabs wandering or waiting for customers. We omitted trajectories with fewer than 5 locations and trajectories with some wrong locations (detected by computing the speed between two consecutive locations). The resulting dataset contains 7.265 trajectories and 60.628 locations.

Table 1 - Utility metrics of a dataset anonymized with different methods

Method	# Trajectories	# Locations	Average distance straight line (km)	Avg. random location entropy	Avg. uncorrelated location entropy	Avg. visits per location
Original dataset	**7.265**	**60.628**	**3,834**	**2,130**	**1,449**	**32,701**
SwapMob	7.136	59.532	3,850	2,511	1,704	40,549
Microaggregation	7.265	60.619	2,480	3,095	2,118	41,40
SwapLocations	7.218	59.083	4,448	3,071	2,107	47,453

4. User manual

The developed package provides a command line interface (CLI) that lets users anonymize a mobility dataset and compute some utility measures over both the original and the anonymized datasets in a straightforward way.

```
$ python -m mob_data_anonymizer
```

4.1. Anonymization methods

The parameter values to configure the anonymization methods are provided to the application using a JSON file:

```
$ python -m mob_data_anonymizer anonymize -f parameters_file.json
```

There are some common parameters to all the anonymization methods:

Table 2 - Common parameters to all anonymization methods

Parameter	Description
method	`Type: string` The name of the anonymization method to be executed. Must be one of the following: - SwapMob - Microaggregation - SwapLocations
input_file	`Type: string` The dataset to be anonymized
output_folder	`Type: string, optional` Folder to save the generated output datasets
main_output_file	`Type: string, optional` The name of the anonymized dataset
save_preprocessed_dataset	`Type: boolean, optional` True: Export the pre-processed dataset
preprocessed_file	`Type: string, optional` The name of the pre-processed dataset

Each of the anonymization methods has some specific parameters that have to be added to the parameters file:

Table 3 - Specifics parameters to be added to the parameters file

SwapMob

Parameter	Description
spatial_thold	`Type: int` Maximum distance (in meters) to consider two locations as close
temporal_thold	`Type: int` Maximum time difference (in seconds) to consider two locations as coexistent

min_n_swap	`Type: int, optional`
	Minimum number of swaps for a trajectory for not being removed.
	`default: 1`

seed	`Type: int, optional`
	Seed for the random swapping process
	`default: None`

JSON config file example for the SwapMob method:

```
{
  "method": "SwapMob",
  "input": "examples/data/cabs_dataset_20080608_0700_0715.csv",
  "output_folder": "examples/output/",
  "main_output_file": "output_SwapMob.csv",
  "save_preprocessed_dataset": true,
  "filtered_file": " preprocessed _SwapMob.csv",
  "spatial_thold": 100,
  "temporal_thold": 60
}
```

Figure 9 - JSON Config file example for the SwapMob method

Microaggregation

Table 4 – Microaggreation parameters

Parameter	Description
k	`Type: int` Minimum number of trajectories to be aggregated in a cluster
lambda	`Type: float` Computing parameter λ (see section 3.1.2) is usually costly. You can use a pre-computed value. If $\lambda = 0$, the temporal component will not be considered. Use this option if all locations of the dataset are from a short time interval. If the field is not included, the parameter λ will be computed.

JSON config file example for the Microaggregation method:

```
{
  "method": "Microaggregation",
  "input_file": "examples/data/cabs_dataset_0700_0715.csv",
  "output_folder": "examples/outputs",
  "main_output_file": "anonymized_Microaggregation_CLI.csv",
  "save_preprocessed_dataset": true,
  "preprocessed_file": "preprocessed_dataset_CLI.csv",
  "k": 3,
  "lambda": 0
}
```

Figure 10 - JSON config file example for the Microaggregation method

Swap Locations

Table 5 - Swap locations parameters

Parameter	Description
k	`Type: int` Minimum number of locations of the swapping cluster
min_r_s	`Type: int` Minimum spatial radius of the swapping cluster (in meters)
max_r_s	`Type: int` Maximum spatial radius for building the swapping cluster (in meters)
min_r_t	`Type: int` Minimum temporal threshold for building the swapping cluster (in seconds)
max_r_t	`Type: int` Maximum temporal threshold for building the swapping cluster (in seconds)

JSON config file example for the Swap Locations method:

```
{
  "method": "SwapLocations",
  "input": "examples/data/cabs_dataset_20080608_0700_0715.csv",
  "output_folder": "examples/output/",
  "main_output_file": "output_SwapLocations.csv",
  "save_preprocessed_dataset": true,
  "filtered_file": "preprocessed_SwapLocations.csv",
  "Max_r_s": 600,
  "Min_r_s": 150,
  "Max_r_t": 200,
  "Min_r_t": 10,
  "k": 3
}
```

Figure 11 - JSON config file example for the Swap Locations method

4.2. Utility metrics

As previously mentioned, the anonymization module also includes a tool to compute and compare some utility metrics of original and anonymized datasets. We leverage the well-known scikit-mobility[2] library to compute these utility metrics. To compute some of these measures, the datasets to be compared are previously tessellated.

Again, the parameter values to compute the measures are provided to the application through a JSON file:

```
python -m mob_data_anonymizer measures -f parameters_file.json
```

Table 6 - parameters values to compute measures

Parameter	Description
methods	`Type: array` The name of the measures to compute. Must be some of the following (please, visit the scikit-mobility website for details): - visits_per_location - distance_straight_line - uncorrelated_location_entropy - random_location_entropy - mean_square_displacement
mode	`Type: string` Type of output. As some measures output is a DataFrame a strategy has to be defined. Must be one of the following: - average: computes the average of each DataFrame (from the original and anonymized dataset) and sends it to stdout - export: join and export both output DataFrames (from the original and anonymized datasets) to a single CSV file
original_dataset	`Type: string` Filepath of the original dataset
anonymized_dataset	`Type: string` Filepath of the anonymized dataset

[2] https://github.com/scikit-mobility/scikit-mobility

Funded by the European Union

output_folder	`Type: string, optional` Folder to save the generated output files

JSON config file example:

```
{
  "original_dataset": "examples/output/preprocessed_Micro.csv",
  "anonymized_dataset": "examples/output/output_Micro.csv",
  "mode": "average",
  "methods": [
    "visits_per_location",
    "distance_straight_line"
  ],
  "output_folder": "examples/output/"
}
```

Figure 12 - JSON config file example for parameters values to compute measures

5. Conclusions and plans for the next version

This document presented the anonymization module of the MobiDataLab Transport Cloud prototype and accompanies the demonstrator available at https://github.com/MobiDataLab/mdl-anonymizer.

We plan to add additional features to the module including, but not limited to new anonymization methods providing different privacy guarantees, additional preprocessing and distance computation mechanisms, methods for the protection of real-time mobility data, and generation of synthetic mobility data. Additionally, the next version will include methods to compute the re-identification risk for different types of mobility data.

Bibliography

1. J. Domingo-Ferrer and R. Trujillo-Rasua, "Microaggregation- and permutation-based anonymization of movement data", Information Sciences, Vol. 208, pp. 55-80, Nov 2012, ISSN: 0020-0255.

6.

2. J. Domingo-Ferrer, S. Martínez and David Sánchez, "Decentralized k-anonymization of trajectories via privacy-preserving tit-for-tat", Computer Communications, Vol. 190, pp. 57-68, Jun 2022, ISSN: 0140-3664.

3. A. Gasmelseed, N. Mahmood, “Study of hand preferences on a signature for right-handed and left-handed peoples,” International Journal of Advances in Engineering & Technology, vol. 1 (5), pp. 41-46, 1963.

4. J. Salas, D. Megías and V. Torra, “SwapMob: Swapping Trajectories for Mobility Anonymization”. Privacy in Statistical Databases – PSD2018. Lecture Notes in Computer Science vol. 11126, pp. 331-346. Sep 2018.

5. S. Shang, L. Chen, Z. Wei, C.S. Jensen, K. Zheng, P. Kalnis, “Trajectory similarity join in spatial networks,” in Proceedings of the VLDB Endowment, vol. 10 (11), pp. 1178-1189, 2017.

6. H. Su, S. Liu, B. Zheng, X. Zhou, K. Zheng, “A survey of trajectory distance measures and performance evaluation,” The VLDB Journal, vol. 29, pp. 3-32, 2020.

Funded by the European Union

MobiDataLab consortium

The consortium of MobiDataLab consists of 10 partners with multidisciplinary and complementary competencies. This includes leading universities, networks and industry sector specialists.

@MobiDataLab
#MobiDataLab

https://www.linkedin.com/company/mobidatalab

For further information please visit **www.mobidatalab.eu**

MobiDataLab is co-funded by the EU under the H2020 Research and Innovation Programme (grant agreement No 101006879).