BehavAuth

Human behaviour based authentication for Internet of Things

Open Call Project for TagItSmart: Grant agreement 688061

WP2: D2.2 Final trials and validation report

Deliverable ID:	**D 2.2**
Deliverable Title:	**Final trials and validation report**
Revision #:	**1.0.1**
Dissemination Level:	**Public**
Responsible beneficiary:	QBE
Contributing beneficiaries:	All
Contractual date of delivery:	28/02/2018
Actual submission date:	21/3/2018

Start Date of the Project: 1 September 2017 Duration: 6 Months

European Commission

Contents

Section 1 - Introduction 5

Section 2 - Social interaction evaluation 6

2.1 Experimental setup 6

2.2 Performance metrics 6

2.2.1 Accuracy 6

2.2.2 Confusion matrix 6

2.3 Results 7

Section 3 - Authentication system evaluation 9

3.1 Experimental setup 9

3.1.1 Dataset 9

3.1.2 Data pre-processing 9

3.1.3 Class distribution 10

3.2 Performance metrics 12

3.2.1 Accuracy 12

3.2.2 Confusion matrix 12

3.2.3 Other performance metrics 13

3.3 Results 13

3.3.1 Social interactions as authentication component 14

3.3.2 Visited locations as authentication component 16

3.3.3 Speed in the environment as authentication component 18

3.3.4 Distance in the environment as authentication component 20

3.3.5 Evaluation of the system as a whole 23

Section 4 - User experience evaluation 26

4.1 Delay in the training process 26

4.2 Delay in authentication 26

4.3 Power consumption and CPU load 27

4.3.1 Experimental setup 27

4.3.2 Performance metrics 27

4.3.3 Results 27

Section 5 - Conclusion 30

Section 6 - References 32

Document history

Revision	Date	Modification	Authors
1.0	28/02/2018	Draft ready for review	N. Palaghias
1.01	21/03/2018	Addressing comments from reviewers	N. Palaghias

Introduction

The main objective of this document is to provide the information related to the trials and the validation activities performed on behalf of the BehavAuth and TagItSmart projects as well as the outcome of these evaluations.

Section 1 provides an introduction to the deliverable about the trials and validations activities conducted during the BehavAuth project. A brief summary of the trials is provided at the beginning of the section describing the trials and experiment performed in order to evaluate first the social interaction detection mechanism and then to evaluate the different authentication components that were developed in the BehavAuth project, leveraging various behavioural cues.

In **Section 2**, an analysis of the evaluation performed for the social interaction detection mechanism that was developed as part of the BehavAuth project is provided. The evaluation was performed in different indoor environments including meeting rooms and corridor. Users were placed in vicinity and were left freely to interact while a human observer was logging the ground truth of the experiments. The ground truth of the experiment was important to log in order to be able to calculate the performance of the social interaction detection mechanism.

Next, **Section 3** described the evaluation of the different authentication components that were developed as part of the BehavAuth project. The evaluation includes 10 different users and each of the authentication components that were developed are individually evaluated to understand their accuracy. The evaluation was performed on the same dataset and included performance analysis of the authentication process based on the social interaction detection mechanism, on location patterns, on average speed patterns of the user in the environment and average distance patterns of the user in the environment. The evaluation was performed using multiple performance metrics including accuracy, confusion matrices and other performance metrics such as precision, recall, f1-score and support.

Finally, this deliverable ends with the conclusion provided in **Section 4**.

Section 1 - Introduction

This constitutes an introduction to the deliverable about the final trials performed in order to evaluate the BehavAuth tool and its components. One of the initial main tasks of the BehavAuth was to develop and evaluate a social interaction detection tool. Then given the social interaction detection data and other contextual data such as location, speed in the environment and distance within the environment, authentication components were developed to provide a behavioural authentication system to the TagItSmart platform.

The social interaction detection tool was focused on leveraging state-of-the-art sensors that are widely available on commercial-off-the-shelf smartphones and develop an intelligent mechanism that will be able to recognise the social interaction in which the user is taking part. The mechanism that was developed focused on leveraging the Bluetooth signal strength and through this, learning and understanding the distance among the users. This distance estimation is dedicated on face-to-face interactions were users have a line-of-sight contact, thus the system does not require any relative orientation estimation to understand if the users are actually facing each other or not. Extensive data collection and feature extraction processes allowed the training of robust machine learning models that were able to accurately estimate the interpersonal distance and further the existence of a social interaction.

Having estimated the existence of a social interaction, authentication components were developed to leverage contextual information such as existence of social interaction, location, speed and distance within the environment in correlation with the time factor. The social interactions that were generated by the corresponding learning model were further used to train an authentication component that leverage the social interaction information including time and users id to understand the social interaction patterns and authenticate the user based on them. Similar the location patterns were used together with the time and instantaneous speed were also used to learn the location patterns of the user and authenticate them based on them. Also, the average speed patterns were used to train machine-learning model that was used to learn and authenticate based on these patterns. Lastly, the average travelled distance patterns were used to train another authentication component that allowed the authentication of the user.

These components were developed in D1.2 [1] and this deliverable focuses on evaluating each of the components in order to understand their accuracy and robustness in real-world environments with actual users.

Section 2 - Social interaction evaluation

This section provides a detailed description of the evaluation of the social interaction detection system using off-the-shelf smartphones. The evaluation took place in a normal office environment with a limited number of users in order to understand the performance of the social interaction detection mechanism as well as any potential drawbacks. This initial evaluation provided feedback to the overall system in order to improve any detected potential drawbacks.

2.1 Experimental setup

The evaluation of the social interaction detection mechanism was designated through performing several experiments in real-world indoor environments. In each experiment, 5 people were participating in the age range of 25-30 and height range 1.65m-1.96m. Each participant was given an off-the-shelf mobile phone i.e. HTC One S which they placed in one of their trousers pockets. Among the indoor environments that were included are: a) meeting room, b) corridor. In each of these places, people were interacting while divided in two groups: a) two participants, b) three participants. Initially, each participant was having a mobile phone in trousers pocket, which was running the social interaction detection application. Then, participants, whilst standing in close proximity, were divided in the afore mentioned groups, and started interacting. The interactions were logged by a human observer in order to establish ground truth. After 10 minutes, participants changed groups and started interacting while in close proximity. In overall, the benchmarking was performed in two group scenarios in a meeting room and in one group scenario in a corridor.

After the end of the three experiments, we analysed the resulted data by inferring in one approach the interpersonal distance detection to understand if users were in proximity to interact or not.

2.2 Performance metrics

The performance metrics used to evaluate the social interaction detection mechanism are focused on the accuracy and the confusion matrix.

2.2.1 Accuracy

The accuracy is a performance metric often used in the machine-learning area in order to understand the overall performance of the approach. It constitutes the fraction of the correctly inferred estimations divided by the overall number of estimations. It is expressed in percentage in order to allow the readers to compare the different types of accuracy in case the comparison is among elements with different number of samples. For example, in the experimentation phase, there are three different experiments that took place with similar duration time however the number of samples that were collected is different for each experiment. For that reason, the accuracy is represented in percentage and not as actual number of estimations.

2.2.2 Confusion matrix

The next performance metric that was used in order to evaluate the social interaction detection mechanism was the confusion matrix. The confusion matrix is a performance metric that provide a more detailed comparison of the error occurrence of the approach. In particular it

analyses the error that is occurring in the true positives, true negatives, false positives and false negatives. The numbers included in this confusion matrix is represented in percentages to be able to compare among three different experiments.

2.3 Results

The participants of the experiments were placed in different indoor environments of an office building and were provided a smartphone with deployed the data collection and social interaction inference app. Participants were socially interacting in different environment in a close distance. In particular the participants were placed in the same environment every time where an experiment was taking place and also they were placed in the vicinity of each other even though they were not interacting. This was performed in order to understand if there is a need to incorporate some kind of relative orientation mechanism in order to further improve the accuracy of the social interaction estimations or if the accuracy achieved by the social interaction detection mechanism is good enough.

The results of the above described experiments led to the computation of the confusion matrices are shown in Table 2.1.

Table 2.1. Confusion matrix for evaluation of the social interaction detection in percentages (%).

	Meeting room 1		**Meeting room 2**		**Corridor**	
	Positive	**Negative**	**Positive**	**Negative**	**Positive**	**Negative**
Positive	83	17	65	35	82.3	17.7
Negative	36	64	60.6	39.4	43.6	56.4

Table 2.1 presents the results of the evaluation of the social interaction detection mechanism through a confusion matrix in order to understand the error occurrence in the different environments where the users were placed in order to socially interact. An overall observation of the social interaction mechanism indicates a quite high accuracy in the process of correctly understanding socially interactions only through the proximity estimation mechanism. This can be observed through the true positives were the social interaction detection mechanism detected with 83% in meeting room 1, 65% in meeting room 2 and 82.3% in the corridor, correct social interactions. The second environment showcased a small decrease in the accuracy of the estimations.

Following, as mentioned at the beginning of this evaluation process, during the experiments users were placed in close distance to each other even though they were not interacting in order to understand if the interpersonal distance estimation model was coherent enough to differentiate among these situations. The result of the close interpersonal distance and the ability of the social interaction detection mechanism to cope with the fact that even though users are in close interpersonal distance they may not socially interact, is provided by the true negatives of Table 2.1. The true negatives indicate the accuracy of the socially interaction detection mechanism in understanding when the users were in such proximity that they were not interacting. Theoretically, in order for the users not to socially interact, they need to be in a longer distance than 3.5 meters [10]. However, as mentioned all users were in proximity to

interact so theoretically the social interaction detection mechanism should provide all of the users that interact with each other. The human body absorption constitutes the key factor that will enable the social interaction detection mechanism to correctly infer which participants are socially interacting with each other and which are not. As the human body contains over 80% of water when the Bluetooth signal is passing through the human body, the signal strength is reduced due to the high absorption. Table 2.1 indicates that in the first case the social interaction mechanism could correctly identify with 64% which users that were in proximity did not socially interact in meeting room 1. In meeting room 2, the accuracy degraded where the recognition took place with 39.4% and in corridor the accuracy increased again to 56.4%.

The results of this evaluation indicated the interpersonal distance estimation technique could accurately detect social interactions among people by leveraging Bluetooth RSSI. In close interpersonal distances the error of false positives is increasing as in different cases the human body absorption is not enough to modify the Bluetooth RSSI signal enough to indicate the social interaction is not taking place. A mechanism that provides the relative orientation of the users could provide additional information in the inference process however it would also introduce a large computational and complexity burden to the data collection and inference process. Thus, it constitutes a trade-off depending on the requirements of the application. In the authentication case, as there are multiple behavioural cues extracted the social interaction detection mechanism is considered sufficient given the accuracy and the computational complexity introduced to the system.

Section 3 - Authentication system evaluation

This section describes the evaluation performed for the authentication system by analysing and evaluating each individual authentication component. The section begins through the experimental setup detailing the information about the dataset, the pre-processing and also the class distribution. Then the performance metrics used in the evaluation process are discussed followed by the results of the experimentation for each of the authentication components.

3.1 Experimental setup

The experimental setup section describes the data collection and data processing that was performed in order to enable the evaluation of the authentication components.

3.1.1 Dataset

For the evaluation of the authentication components a state-of-the-art dataset was used. This dataset was selected for a number of reasons including that it provided the required number of users (10 users), the duration of the experiment that lasted 10 weeks and also the fact that the dataset included all the required data for the BehavAuth system. Among the students who completed the study, are undergraduates and graduate students. In terms of gender, the dataset included both males and females. The participants that were involved in the study were enrolled in the CS65 Smartphone Programming class, a computer science programming class at Dartmouth College that is offered to both undergraduate and graduate students during Spring term in 2013. All students were offered an unlocked Android Nexus 4s to complete assignments and class projects. Each mobile phone had deployed a data collection app that was monitoring the selected sensors and without any interaction with the user, the mobile phones were uploading the data to the backend server. The data off-loading was performed only when the device was charging and only through the WiFi interface. The uploaded data were monitored through scripts to understand if the users are using the phones and participating in data collection.

3.1.2 Data pre-processing

The StudentLife dataset [4] provided by Dartmouth University is a quite large and extensive dataset that included various types of sensor and questionnaire data. In order to be able to leverage the collected data in the BehavAuth project, there was a need to perform some pre-processing of the data, before applying the machine-learning models required for the authentication process.

The data pre-processing procedure required the following steps:

1. **Dataset**. Initially there was a need to extract only the sensor data related to the BehavAuth system. This extraction refers to the Bluetooth and Location measurements for the first 10 users.

2. **Labelling the data**. As the data for the Bluetooth and the Location measurements for each user are provided in individual file, there was a need to label the data with the ID of the user. In particular, another column was added to the dataset describing the ground truth which is the ID of the user, to whom the data belong. In this way, we will be able to analyse the results provided by the machine-learning algorithms in order to

understand the performance of the system by comparing the outcome of the algorithms with the actual value.

3. **Additional features**. The StudentLife dataset includes various sensor and questionnaire data in a digital form. However, it does not include some specific features such as the speed and distance within the environment that were mentioned in D1.1 [2]. The location dataset includes a field "speed" which refers to the instantaneous speed of the user the moment the location measurement is collected. As there are many points in the dataset where the speed is 0, so there is a need to calculate a more indicative measurement of the speed of the user in the environment. For that reason, the location data combined with the timestamps of these location were used in order to compute the users' speed and distance within the environment.

    a. **Distance within the environment**. The distance within the environment refers to the average distance the user moves within the environment every period of the location sampling frequency. The distance was calculated using the current and the previous location measurements represented in latitude and longitude. As latitude and longitude are in degrees there is a need to apply special techniques for calculating the distance. The initial approach would be to use Euclidean distance; however, this does not take into account that the measurements are in degrees and also that the earth is a sphere. Another solution is Haversine formula [5], which considers that the earth is a sphere but according to literature [6] results in 0.5% error. For the above reasons, BehavAuth focused on using Vincenty [7] which provides more accurate ellipsoidal models. The distance is represented in kilometres and refers to the average distance between two sequential location measurements.

    b. **Speed within the environment**. The average speed is provided by a general form velocity = distance / time. It was assumed that each user started to use the mobile data collection app from stance, where velocity was 0. So, the initial speed sample for each user is 0. Then, for each sequential pair of locations and timestamps, the distance and time were calculated. The distance was calculated similarly with the (a) point in kilometres. The time was calculated by subtracting the newest timestamp minus the previous timestamp – both timestamps were represented in Unix time – which resulted in the time in seconds and was converted into hours. Using the initially mentioned formula the speed was calculated and provided an average speed of the user in the environment.

4. **Merging the datasets**. Having extracted the appropriate data, labelled them with the ground truth to be able to calculate various performance metrics and added the additional features required to extract the appropriate information, the various small datasets were merged into one concrete data for social interactions and one for location/speed/distance. These two datasets were not further merged as the number of samples of each of dataset is different, given the different sampling frequency of each of the sensors (Bluetooth and Location/GPS/WiFi/Cellular).

3.1.3 Class distribution

In machine-learning it is important to take into account the skewness observed in the different classes that are involved in the classification process. A class is the type of data that the

algorithm infers and provides it as a result of the classification process. In this evaluation, the class is the user ID that the BehavAuth mechanism is trying to infer by leveraging sensor data. The class has 10 different values as there are 10 different users involved in the classification/authentication process. In the ideal situation, for each user there would be the same number of samples involved in the authentication process. However, as the dataset includes real-world data from students at the University of Dartmouth, it is logical that for the same duration, different number of samples were collected. This is because the sensors that provide the data are not operating fully periodically in order to minimise the poqwe consumption of the smartphone.

For example, a user during the night is sleeping and has left the smartphone on the table. While the user is sleeping, there is no change in the data that the sensors provide. So, to minimise the power consumption the app discontinues the sensing process. For this reason, the number of samples for each user is different including also for each sensor. For example, even though the user is on the same location, other users with Bluetooth interface could be in the range and can pass by. This means that various sensor measurements will be created in this case. So, the Bluetooth and the location interface will create different number of samples.

In the next two subsections, the class distribution for each of the 10 users involved is detailed.

3.1.3.1 Bluetooth

This subsection presents the class data distribution in terms of number of samples with respect to the Bluetooth dataset. As mentioned in previous section, the Bluetooth and Location dataset have a different sample distribution as the sampling frequency of each of the sensors differs with each other. In the case of the Bluetooth sensor, the non-equal class data distribution happens because the data are collected in an opportunistic manner. This means that the data are generated based on the context in which the device is placed. For example, if the device is placed in an environment where multiple Bluetooth devices are discoverable, then every periodic scan will generate multiple Bluetooth samples for the same timestamp.

Table 3.1. Class Data distribution in terms of number of samples.

0	1	2	3	4	5	7	8	9	10
7321	6542	18782	2804	7372	59414	9159	18098	13840	16079

Table 3.1 provides the class distribution of the Bluetooth dataset in terms of number of samples. It can be noticed that the number of samples leveraged in every class is different. The reason was explained in the previous paragraph and is related to the high dependence of the environment of the user. It should be noted that the duration of the experiment is the same in all cases. For users 2, 8, 9 and 1 the number of samples is quite increased in comparison to the users 0,1, 3,4 and 9.

3.1.3.2 Location

This subsection presents the class data distribution in terms of number of samples with respect to the Location dataset. It is observed that the location dataset has a class data distribution with differences in the number of samples from each class. However, a smaller difference among the different classes is observed in the location dataset in comparison to the Bluetooth dataset. As mentioned, the Bluetooth dataset and especially the number of samples collected are highly dependent on the environment in which the data collection process is taking place. In contrast, the location data collection does not have such a high dependence on the environment. There are though cases where the environment and the user could affect the number of samples collected. For example, if the user turns off the location services of the

mobile phone then the data collection will be paused and a gap in the data will be observed. Also, in case the user configures the smartphone in a certain way that disables the fused location sensor, then a gap in the location data will observed. The particular configuration requires that the user is indoors (the GPS sensors does not operate indoors), the mobile phone is on airplane mode (WiFi and cellular are disabled). In this case, the smartphone is not able to retrieve a new location measurement.

Table 3.2. Class Data distribution in terms of number of samples.

0	1	2	3	4	5	7	8	9	10
4316	5541	6607	3502	3620	5388	3105	6192	5062	4457

Table 3.2 provides the class distribution of the Location dataset in terms of number of samples. It can be noticed that the number of samples leveraged in every class is different but still close to each other in comparison to the Bluetooth class distribution. The location sensing process has a constant sampling frequency. However, in order to provide a more power preserving approach the location sensor combines multiple sensors such as GPS, WiFi and cellular in order to provide a location estimation. This sensor fusion mechanism also considers the lack of movement also. This means that in cases where the device is idle and is not moved, in order to reduce the power consumption of the fused location sensors, the location interface does not provide any sample. The sample difference in the location data occurs due to the fact that as users have different behaviour in their lives, then different number of samples are produced. Finally, it could be noted that all the number of samples are between 3500-6200 samples.

3.2 Performance metrics

This section provides a description of the performance metrics used in order to evaluate the authentication components. This description focuses on enabling the reader to understand the different performance metrics. This will allow the reader to understand the evaluation results presented in the next subsection about each of the different authentication components. The performance metrics [9] used in the study are (a) accuracy, (b) confusion matrices and (c) other performance metrics such as precision, recall, F1-score and support. For the accuracy of the algorithms 10-Fold cross validation was applied and for the rest (b) and (c) 20% of the dataset was considered as testing dataset.

3.2.1 Accuracy

Accuracy constitutes a metric of systematic errors and a measure of statistical bias. It is computed through the fraction of correct predicted estimations divided by the overall number of estimations. It is computed at the end of the evaluation process where the overall number of correct and incorrect estimations are known. This performance metric indicates the overall performance of the algorithm with respect to the whole dataset. It does not provide any indication on which class or cases the misclassification is happening and where is the source of error. It does however provide an overall estimation of the error of the approach.

3.2.2 Confusion matrix

The confusion matrix or error matrix constitutes a matrix that includes all the classification results of the dataset used. The dimensions of the classification matrix are dependent on the number of classes defined in the classification process. For example, in this evaluation as the number of different classes is 10 (is the number of users), the dimensions of the confusion matrix will be 10x10. The confusion matrix will indicate the number of errors and correct estimations in the classification process. The diagonal of the matrix includes all the correct

estimations of the algorithms. So, in the ideal case all of the non-diagonal elements should be zero. The smaller the non-diagonal numbers are the better. This allows the reader to understand among which users is the misclassification taking place and then by viewing the data the algorithm design can improve the performance of the algorithm by finding more informative features that will tackle the misclassification observed at these places of the matrix.

It should be noted that the confusion matrix refers is the outcome of the classification of one particular algorithm. The initial step is to evaluate different algorithms and compare them based on their accuracy, which constitutes an overall performance metric of each of the algorithms. Then, by concluding on the particular algorithm the confusion matrix is used to better understand where the different errors are occurring.

3.2.3 Other performance metrics

This subsection includes some other performance metrics that provide information about the performance of the different authentication components. These performance metrics are grouped together in this subsection in order to keep consistency in the presentation of the results that follow in Section 3.3. These performance metrics are related to one specific algorithm. In the evaluation section, the results related to these performance metrics are related to one algorithm in order to understand better the performance of the particular algorithm.

3.2.3.1 Precision

Precision or positive predictive value is the fraction of the true positives divided by the sum of true positives and false positives. In other words, it is “the fraction relevant instances that have been retrieved over the total amount of relevant instances”. Precision is a measure of result relevancy. A high result in the precision of an algorithm indicates a low false positive rate.

3.2.3.2 Recall

Recall is defined as the fraction of the number of true positives divided by the number of false negatives. Recall is a measure of how many truly relevant results are returned by the algorithm. A high result in the recall of an algorithm indicates a low false negative rate.

3.2.3.3 F1-score

The F1-score constitutes a combination of the precision and recall in order to provide an overall performance metric of the system. In particular, the F1-score is defined as the harmonic mean of the precision and the recall of an algorithm. In case both precision and recall are returning accurate then the algorithm gets close to an ideal system.

3.2.3.4 Support

Support is the number of samples used in the evaluation process from each class. The evaluation process considered 20% of the overall dataset as testing dataset. The support performance metrics indicates exactly how many samples where used in the testing dataset from each of the different classes.

3.3 Results

This section provides the results of the evaluation of the different authentication components. The evaluation was performed through the dataset described in Section 3.1 in the Experimental Setup. Also, the evaluation metrics used throughout the evaluation in order to understand the performance of the algorithms were presented in Section 3.2. The evaluation of the different components is presented in the following order, based on the authentication

models developed for understanding the different behavioural cues: (a) social interactions, (b) location, (c) speed and (d) distance.

Regarding the performance metrics, for each of the behavioural cues authentication model the evaluation follows a certain sequence. Initially, different machine-learning techniques are evaluated in order to understand which algorithm provides the best fitting for authenticating the user based on the different behavioural data. Then, the algorithm with the highest accuracy is selected and further evaluation is performed. This additional evaluation tries to split the model in pieces in order to understand the pros and cons of the model. In particular, the confusion matrix of the machine learning algorithm is providing a more detailed analysis of the misclassification occurring through the particular model. Also, the other performance metrics such as precision, recall, f1-score and support are provided to understand the robustness of the model.

3.3.1 Social interactions as authentication component

The first model that will be evaluated is the social interaction based authentication component. This component leverages social interactions detected in users' daily routines in order to understand the identity of the user and authenticate him/her as the owner of the device.

3.3.1.1 Accuracy

The initial evaluation process of the authentication component based on social interactions, was related to the different machine-learning algorithms. This would provide an understanding of the accuracy achieved by performing authentication based on the social interactions of the users. The machine-learning methods used were Linear Regression (LR) [11], Linear Discriminant Analysis (LDA) [12], K-Nearest Neighbour (KNN) [13], Decision Tree Classifier (DTC) [14] and Gaussian Bayes (GB) [15]. The results of the evaluation are present in Table 3.1.

Table 3.3. Accuracy of the models based on different machine-learning models.

	Linear Regression	**Linear Discriminant Analysis**	**K Nearest Neighbour**	**Decision Tree Classifier**	**Gaussian Bayes**
Accuracy	0.384801	0.399597	0.641003	0.793345	0.401698

The best performance in terms of overall accuracy was achieved by the Decision Tree Classifier with 79% accuracy. The second best model was trained given the KNN with 64% accuracy. The other three models that were trained achieved similar accuracy (38-40%) having a quite poor performance. This indicates that their performance is worse than a random classifier that will guess randomly the result of the inference. The KNN considering the distance calculation managed to achieve an accuracy of 64%, which as individual component is considered low and cannot be used as an authentication component. The DTC managed to outperform the rest of the algorithms and for that reason was selected as main classification model for the social interaction component.

3.3.1.2 Confusion matrix

Having selected the DTC model that outperformed the other machine-learning techniques, a more in depth evaluation was performed to understand the error occurrence of the algorithm. In particular, the model was trained based on the DTC machine learning technique and 20% of the dataset was left out as a testing dataset. Running the DTC model on the 20% testing dataset provided the confusion matrix provided at Table 3.4.

Table 3.4. Confusion matrix for the decision tree model.

ID	0	1	2	3	4	5	7	8	9	10
0	**879**	24	118	16	28	125	36	79	53	81
1	23	**961**	31	4	13	157	14	19	62	62
2	127	25	**2923**	13	91	175	57	101	122	146
3	10	2	8	**417**	17	18	9	28	16	17
4	47	9	93	14	**902**	121	29	78	102	112
5	113	167	203	20	96	**10651**	32	145	210	197
7	55	13	42	8	26	33	**1589**	33	31	37
8	70	21	107	39	67	116	40	**2837**	63	224
9	59	49	142	15	84	214	33	74	**1921**	140
10	57	43	126	11	109	241	28	251	128	**2255**

Table 3.4 describes the confusion matrix of the DTC model trained for authenticating users based on the detected social interactions. Across the diagonal are the numbers of the correctly predicted users. It should be taken into account that these are the actual correctly predicted numbers and that the total number of estimations for each user is different. So, for example, even though the user 3 indicates very low misclassification numbers, the overall number of predictions is 542 for that user, which is proportionally low in comparison to user 5 that received 11834 predictions. While observing users 0, 4 and 10, and given the number of samples provided for their class it could be inferred that these constitute the users with the highest misclassification error. The misclassification error for user 0 is taking place mainly with users 2, 5, 8 and 10. For user 4, the misclassification error is happening with users 3, 5, 8 and 9. For user 10, the misclassification error is happening with users 2, 4, 5, 8 and 9. This error indicates that these people have quite similar social interactions with the people that the misclassification error happens. For user 5, which has the largest number of samples in the evaluation process the error is minimal in comparison to the overall number of samples. For the rest of the users such as 1, 2, 3, 7, 8, 9 the errors are evenly distributed with some increased misclassification error with users 2 and 5.

3.3.1.3 Other performance metrics

After discussing about the confusion matrix for the DTC model for authenticating the users based on the social interaction patterns, the evaluation results continue with the discussion of other performance metrics including precision, recall, f1-score and support. The definitions for each of these performance metrics have been discussed in the above Section 3.2.3. Following, Table 3.5 presents the results of the other performance metrics for the evaluation of the DTC model for authenticating users based on the social interaction patterns.

Table 3.5. Other performance metrics for the decision tree model.

Class	Precision	Recall	F1-score	Support
0	0.61	0.61	0.61	1439
1	0.73	0.71	0.72	1346
2	0.77	0.77	0.77	3780
3	0.75	0.77	0.76	542
4	0.63	0.60	0.61	1507
5	0.90	0.90	0.90	11834
7	0.85	0.85	0.85	1868
8	0.78	0.79	0.78	3584

9	0.71	0.70	0.71	2731
10	0.69	0.69	0.69	3250
Avg / Total	0.79	0.79	0.79	31881

Table 3.5 presents the results of the evaluation of the DTC model for authenticating users based on their social interaction patterns with respect to the precision, recall, f1-score and support. The best performance in terms of precision, recall and f1-score is observed by the user 5. The support (number of class samples) is at least 3 times larger than the other users indicating possible the need for larger number of samples in order to increase the accuracy of the algorithm. Regarding users 1, 2, 3, 8 and 9 the performance of the precision is between 0.71 and 0.78, the performance of recall is between 0.71 and 0.79, and for the f1-score the performance is between 0.71 and 0.78. Similar to what the confusion matrices indicated, the highest error is observed for users 0, 4 and 10. The precision and recall is the same for user 0 (i.e. 0.61) and for user 10 (i.e. 0.69). For user 4 it is observed a slight increase of the precision with respect to the recall resulting in a skewed f1-score towards the recall. Lastly the overall precision, recall and f1-score is kept on the same value which is 0.79. The support values are the corresponding given that the 20% of the dataset was used for testing purposes.

3.3.2 Visited locations as authentication component

Following to the social interaction pattern detection is the use of location patterns as an authentication tool. This component focuses on identifying the various users based on the location patterns that they follow during their daily routines. The representation of the location instances is based on geographical coordinates i.e. latitude and longitude.

3.3.2.1 Accuracy

The first step for the evaluation of the location-based authentication is develop and compare different machine-learning models that leverage the location of the different users in order to identify them as individuals. The different models were derived by leveraging machine-learning techniques including LR, LDA, KNN, DTC and GB. The different models were evaluated based on 10-fold cross validation and based on the same dataset. The results of the evaluation are presented on Table 3.6.

Table 3.6. Accuracy of the models based on different machine-learning models.

	Linear Regression	**Linear Discriminant Analysis**	**K Nearest Neighbour**	**Decision Tree Classifier**	**Gaussian Bayes**
Accuracy	0.138342	0.161234	0.039530	0.890017	0.111161

Table 3.6 provides the results of the LR, LDA, KNN, DTC and GB machine-learning models for identifying different users based on their location patterns. The highest accuracy was achieved by the DTC model that managed to differentiate users with accuracy 89%. The rest of the models had quite poor performance that was less than 20% accuracy. This indicates that all 10 users were in quite similar locations during the experimentation period. So the algorithms could not differentiate among the different users based solely on the location. The DTC managed to differentiate among the different users by creating specific rules for the particular locations and in that sense managed to verify the identities of the 10 different users. For that reason the DTC was selected as machine-learning technique to identify the users based on the location patterns.

3.3.2.2 Confusion matrix

The next step in the evaluation process of the location based authentication is the process of in depth evaluation of the DTC model that was selected through the accuracy results presented in the previous subsection. The machine-learning model based on the DTC classifier was trained based on 80% of the samples of the dataset. The rest 20% of the dataset was kept in order to use it in the testing process. Having trained the model and evaluated the model based on the testing dataset, the confusion matrix presented in Table 3.7 was derived.

Table 3.7. Confusion matrix for the decision tree model.

ID	0	1	2	3	4	5	7	8	9	10
0	**769**	6	15	1	14	6	8	12	9	5
1	8	**1043**	10	0	7	3	3	6	16	7
2	17	15	**1181**	1	11	33	6	18	19	16
3	5	0	0	**710**	1	6	0	3	2	1
4	8	7	21	1	**590**	27	8	20	24	18
5	11	7	24	3	16	**958**	8	20	12	24
7	3	11	17	0	8	7	**599**	12	7	4
8	6	7	31	7	17	26	1	**1027**	23	33
9	10	10	21	6	22	21	7	16	**897**	15
10	8	4	14	1	13	22	5	30	15	**773**

Table 3.7 provides a more detailed indication of the error occurrence for the DTC machine-learning model for identifying individuals based on their location patterns. The smallest errror for person identification is observed on user 3, where with respect to users 1, 2 and 7 the algorithm did not perform any misclassification and for the rest of the users at 728 samples it had in worst case 6 misclassification for one user. The next best performance was achieved for user 1, 7 and user 0, where the misclassification per user reached 16, 17 and 15 samples in 1317, 667 and 845 overall samples. Users 2, 8 and 9 had similar performance with some differentiation in the maximum number of misclassifications per user; it is indicative that the highest error per user is 33, 33 and 22 over an overall number of samples of 1317, 1178 and 1025 accordingly. Users 5 and 10 had similar number of maximum misclassifications as the previous users, 24 and 30 accordingly. Finally the worst performance was observed at user 4 where even though the maximum number of misclassification per user was 27 as the overall number of samples was quite small i.e. 724, the number of errors is quite significant in comparison to other users.

3.3.2.3 Other performance metrics

In order to get more in depth in the performance analysis, there is a need to evaluate the DTC machine-learning model for location-based authentication with additional performance metrics. These additional performance metrics are related to the precision, recall, f1-score and support. These performance metrics will provide and additional view on the performance of the algorithm with respect to the different users. The confusion matrix above focused on understanding the error occurrence and among which users, the machine learning model does not provide the required result. For this section, the discussion about the precision, recall, f1-score and support will allow the reader to understand the robustness of the algorithm with respect to the true positives, true negatives, false positives and false negatives.

Table 3.8. Other performance metrics for the decision tree model.

Class	Precision	Recall	F1-score	Support
0	0.91	0.91	0.91	845
1	0.94	0.95	0.94	1103
2	0.88	0.90	0.89	1317
3	0.97	0.98	0.97	728
4	0.84	0.81	0.83	724
5	0.86	0.88	0.87	1083
7	0.93	0.90	0.91	667
8	0.88	0.87	0.88	1178
9	0.88	0.88	0.88	1025
10	0.86	0.87	0.87	885
Avg / Total	0.89	0.89	0.89	9555

Table 3.8 provides the summary of the evaluation with respect to the precision, recall, f1-score and support. The highest performance in terms of precision, recall and f1-score was achieved while authenticating the user 3, where the performance was 0.97, 0.98 and 0.97 accordingly. The next best performance was observed on user 1 with 0.94, 0.95 and 0.94 scores for precision, recall and f1-score. Following are users 7 and 0, with f1-score at 0.91. Next users 2, 8 and 9 dropped to 0.88 for f1-score and precision as well as a small difference of 0.01 in recall. Then users 5 and 10 achieved 0.87 f1-score and precision 0.86, while user 10 managed 0.88 recall, 0.01 higher than user 5 in recall. Finally, although user 4 had the worst performance, it managed to stay over 0.80 in all the metrics and achieved 0.84 precision, 0.81 recall and 0.83 f1-score. Regarding the support, the number of samples are not equal however it is obvious that the number of samples does not affect the performance of the algorithm, as there are cases like user 7 that had the smallest number of samples and managed all three metrics above 0.90

3.3.3 Speed in the environment as authentication component

The next authentication component that was developed during the BehavAuth, was to leverage the speed patterns of the user in the environment in order to identify the user. The location sensor provides also a measurement of speed that is defined as instantaneous speed of the user. However, in many cases this instantaneous speed was computed as zero because the user was not moving at that particular point in the time. So, in order to be able to derive the speed patterns of the user, a notion of average speed patterns was introduced among the location sampling points. Based on these features initially different machine-learning models were developed in order to understand the accuracy of the different machine-learning techniques used and then after selecting the most accurate and robust solution, additional evaluation was performed to understand the error and the robustness of the approach.

3.3.3.1 Accuracy

The first step in the evaluation of the speed patterns authentication model was to understand the accuracy of the models. For that reason, different machine-learning techniques were used in order to train models that were leveraging the users speed in the environment in order to authenticate users. The machine learning techniques that were used were LR, LDA, KNN, DTC and GB. The evaluation was performed using 10-fold cross validation on the dataset that was used the above sections. The results of the evaluation are provided in Table 3.9 and in the next paragraphs a discussion of the results is provided.

Table 3.9. Accuracy of the models based on different machine-learning models.

	Linear Regression	Linear Discriminant Analysis	K Nearest Neighbour	Decision Tree Classifier	Gaussian Bayes
Accuracy	0.143492	0.148881	0.904949	0.922552	0.143126

Table 3.9 presents the results of the evaluation of the speed patterns models using different machine learning techniques such as LR, LDA, KNN, DTC and GB. After analysing the table, it can be observed that the best accuracy for authenticating users based on speed patterns is achieved again by DTC model with 92%. The next best machine-learning model that was trained is the KNN with 90% accuracy. Both models performed very well in identifying individuals and differentiating among the 10 users. The LR, LDA and GB performed poorly as they achieved around 14% and could not be used to authenticate users based on their speed patterns in the environment. Given the good performance of the DTC and the KNN, even though there is only 2% difference in the accuracy of the two models, the model with the highest accuracy was selected. Thus, the DTC was chosen as a model for authenticating users based on their speed in the environment.

3.3.3.2 Confusion matrix

The evaluation of the accuracy of the different machine-learning models provided an overview of the performance of different machine-learning techniques on authenticating individuals based on their average speed in the environment. This enabled the selection of the machine learning model which was the DTC classifier. To further evaluate the selected classifier, additional evaluation was performed. The evaluation required the splitting of the dataset into 80% training and 20% testing. Furthermore, the DTC classifier was evaluated on the 20% of the dataset and the results were analysed through a confusion matrix to understand the different error occurrence. The confusion matrix for the DTC model for authenticating users based on their average speed in the environment, is provided in Table 3.10.

Table 3.10. Confusion matrix for the decision tree model.

ID	0	1	2	3	4	5	7	8	9	10
0	**821**	1	6	3	2	2	4	3	3	0
1	2	**1046**	6	0	8	3	9	8	17	5
2	6	20	**1229**	4	22	23	12	14	14	17
3	4	1	2	**668**	1	1	5	4	3	0
4	4	10	18	2	**618**	18	9	17	16	9
5	3	1	35	0	9	**999**	1	14	7	7
7	4	1	14	4	10	11	**567**	2	9	3
8	3	15	11	4	25	14	11	**1095**	18	18
9	3	15	12	1	16	4	8	20	**916**	13
10	1	3	19	3	17	10	8	8	13	**833**

Table 3.10 presents the results of the evaluation of the DTC model for user authentication based on their speed patterns. The performance is observed at user 3, where in best case the authentication model did not make any mistake and in worst case performed 5 erroneous estimations for this user in a dataset of 689 samples. Then for user 0 the performance was similar where again in best case the authentication mechanism was perfect and in worst case it made 6 erroneous estimations in 845 samples. Following, for user 1 where the number of samples increases to 1104, in best case the authentication was perfect and in worst case it made 17 mistakes. Next, for users 5, 8 and 10 in the best case the algorithm managed 0, 3

and 1 errors per user and in worst case 35, 25 and 17 for 1076, 1214 and 915 number of samples accordingly. For users 2, 7 and 9 in the best case the algorithm managed 4, 1 and 1 while in the worst case 23, 14 and 20 for 1361, 625 and 1008 number of samples accordingly. For user 4, in the best case scenario the model managed to make 2 errors per user and in the worst case 18 errors, taking into account that the number of samples of the class was 721 samples.

3.3.3.3 Other performance metrics

In order to get a better understanding of the accuracy of the DTC model for authenticating the users based on their average speed in the environment, additional evaluation was performed. The process of the evaluation was similar to the aforementioned in the previous section, where 20% of the dataset was used for testing. The analysis performed in this subsection is leveraging metrics such as precision, recall and f1-score. The results are provided initially in Table 3.11 and are discussed in further detail in the next paragraph.

Table 3.11. Other performance metrics for the decision tree model.

Class	Precision	Recall	F1-score	Support
0	0.96	0.97	0.97	845
1	0.94	0.95	0.94	1104
2	0.91	0.90	0.91	1361
3	0.97	0.97	0.97	689
4	0.85	0.86	0.85	721
5	0.92	0.93	0.92	1076
7	0.89	0.91	0.90	625
8	0.92	0.90	0.91	1214
9	0.90	0.91	0.91	1008
10	0.92	0.91	0.92	915
Avg / Total	0.84	0.84	0.84	9558

Table 3.11 presents the results of the evaluation of the DTC model for authenticating users based on their average speed in the environment, with respect to precision, recall, f1-score and support. The best performance is observed at users 0 and 3 with f1-score 0.97, while user 0 has a small decrease of 0.01 in the precision; recall is the same for both users at 0.97. The user that follows is user 1, with precision 0.94, recall 0.95 and f1-score at 0.94. Next, users 5 and 10 achieved f1-score and precision at 0.92, while their recall was 0.93 and 0.91 accordingly. Following, users 2, 8, 9 with f1-score at 0.91, while their precision and recall varied between 0.90-0.92. The authentication process for user 7 managed f1-score of 0.90. The worst performance was observed at user 4, however the f1-score is at 0.85, similar to precision and the recall was 0.86. Although for user 7, the performance was the worst in comparison to the other users, it still managed a high score. Considering the precision, recall, f1-score and the support, the number of samples included for each class were enough to provide high performance for all the users above 0.85 for precision, recall and f1-score.

3.3.4 Distance in the environment as authentication component

The last authentication component that was developed during the BehavAuth project, is the use of the average travelled distance of the user in the environment, in order to differentiate users and understand the identity of the user. Simple locations that a user has visited may be quite similar in cases for example of two people working in the same environment and living in the same building. However, the distances that they travel during their daily lives are different

as people are differently active among each other. This component focused on providing and detecting the average distance patterns of the user in the environment during their daily lives. The first step towards developing and evaluating the authentication component based on the average distance travelled in the environment, is to evaluate the accuracy of different machine-learning models. The different machine-learning models use the average distance a user is travelling and focus on inferring the identity of the user by differentiating among the mobility patterns of other users.

3.3.4.1 Accuracy

The initial step towards evaluating the authentication mechanism based on the average distance a user has travelled is to develop and evaluate various machine-learning models. The different machine-learning models will provide an overview of the ability of the machine-learning to cope and solve such a problem. In order to perform such evaluation, different machine-learning models were developed using different techniques such as LR, LDA, KNN, DTC and GB. The models were developed and evaluated on the same dataset, while 10-fold cross validation technique was used in each of these models. The results of this evaluation are provided in Table 3.12 and also a discussion about the results is provided in the paragraph below.

Table 3.12. Accuracy of the models based on different machine-learning models.

	Linear Regression	**Linear Discriminant Analysis**	**K Nearest Neighbour**	**Decision Tree Classifier**	**Gaussian Bayes**
Accuracy	0.140170	0.145977	0.942116	0.983836	0.111687

Table 3.12 presents the results of the evaluation of the different machine-learning models developed, leveraging the average distance that the users are travelling in their daily lives. Given the results of the evaluation, the best performance was observed by the DTC and the KNN, which both managed accuracy over 90%. In particular, the highest accuracy was achieved by the DTC with 98% and the next best performance was observed by KNN with 94%. The performance of the rest of the models, meaning LR, LDA and GB is poor below 20%. In particular, the LR and LDA managed an accuracy of around 14% while the GB model managed only 11%. This indicates that only the KNN and DTC machine-learning techniques were able to differentiate among the 10 users with a reasonably good accuracy (over 90%). It should be mentioned that through the rules defined by the DTC, the model presented only 2% error. KNN performed also very well by managing only 6% error. This indicates the significance of the average travelled distance as a feature for the authentication process. Given that the DTC showcased the lowest error in authenticating users, that machine-learning technique was selected for the continuation of the evaluation of the authentication component based on the average travelled distance.

3.3.4.2 Confusion matrix

The previous section described the development and evaluation of different machine-learning models leveraging the average travelled distance in the environment during peoples daily lives. The evaluation provided the outcome that the DTC model had the smallest error with respect to 10 users monitored by 10 weeks. For that reason, the DTC was selected as a machine-learning technique to train the authentication model. In order to further evaluate the authentication model based on the average travelled distance, additional performance evaluation was conducted. In this case, a DTC model was trained focused on the average distance travelled. For training the model, 80% of the dataset samples were used for training

the machine-learning model and 20% of the samples were used for testing the developed model. The results of the evaluation are presented through a confusion matrix in Table 3.13.

Table 3.13. Confusion matrix for the decision tree model.

ID	0	1	2	3	4	5	7	8	9	10
0	**844**	0	1	0	0	0	0	0	0	0
1	1	**1083**	1	0	0	3	1	2	13	0
2	0	1	**1346**	1	1	2	4	0	6	0
3	0	0	0	**678**	1	6	0	2	2	0
4	0	3	1	0	**702**	0	3	2	3	7
5	0	4	1	5	0	**1063**	1	1	1	0
7	0	1	2	0	5	0	**615**	1	0	1
8	0	8	0	1	10	0	5	**1189**	0	1
9	1	4	2	5	3	1	1	2	**987**	2
10	0	1	0	0	4	0	1	2	2	**905**

Table 3.13 presents the results of the evaluation of the authentication component based on the average travelled distance and using the DTC machine-learning technique. An overall view of the confusion matrix indicates the high performance of the algorithm due to the small number of erroneous estimations. The best performance is observed at user 0, where in 845 number of estimations, only 1 estimation is erroneous; all users except from one (user 2) where correctly authenticated in all cases. Following, users 2, 5 and 10 were estimated with high accuracy with minimum error estimation per user was 0 for all three and maximum 6, 5 and 4 accordingly. Next users 1, 3, 7, 8 and 9 had minimum erroneous estimations per user 0 (except user 9 that had minimum error 1 estimation) and maximum erroneous estimations 13, 6, 5, 10 and 5 accordingly. In comparison to the rest of the users, the lowest accuracy was managed by user 4 where the minimum erroneous estimations per user is 0 and the maximum is 7. The performance categorisation that was performed above takes into account the overall number of samples from each different class (i.e. from each different user) that were used in the evaluation process.

3.3.4.3 Other performance metrics

The next step in the evaluation process of the authentication component based on the average travelled distance in the environment is to use state-of-the-art performance metric for machine-learning techniques to understand the robustness of the algorithm. In the previous step, the confusion matrix of the DTC model was presented and analysed. In this section, the same model is evaluated by considering the precision, recall, f1-score and the support. The evaluation uses the same setup as the previous section, where 80% of the data samples are used for training and 20% of the data samples are used to evaluate the model. The following Table 3.14 provides the results of the evaluation of the DTC model based on the average distance and then the next paragraph provides the analysis of the table.

Table 3.14. Other performance metrics for the decision tree model.

Class	Precision	Recall	F1-score	Support
0	1.00	1.00	1.00	845
1	0.98	0.98	0.98	1104
2	0.99	0.99	0.99	1361
3	0.98	0.98	0.98	689

4	0.97	0.97	0.97	721
5	0.99	0.99	0.99	1076
7	0.97	0.98	0.98	625
8	0.99	0.98	0.98	1214
9	0.97	0.98	0.98	1008
10	0.99	0.99	0.99	915
Avg / Total	0.98	0.98	0.98	9558

Table 3.14 present the results of the evaluation of the DTC with respect to other performance metrics such as precision, recall, f1-score and support. The best performance that was achieved was observed at user 0, where all three metrics, precision, recall and f1-score are at the highest possible value, 1.0. The next best, are users 2, 5 and 10 with 0.99 f1-score, recall and precision. Following, with 0.98 f1-score are the users 1, 3, 7, 8 and 9. Out of them, the users 1 and 3 achieved 0.98 precision and recall. Then the users 7,8 and 9 achieved 0.98 recall however their precision was 0.97, 0.99 and 0.97 accordingly. Finally, user 4 relatively to the other users achieved the lowest performance in terms of precision (0.97), recall (0.97), and f1-score (0.97) but still the numbers are high. In overall, the performance of the algorithm in terms of precision, recall and f1-score is very good. The support numbers differ among each other however as the results show the number of samples did not affect the performance of the algorithm.

3.3.5 Evaluation of the system as a whole

The next step in the process of evaluating the performance of the BehavAuth system is to evaluate the system as a whole. This means that the evaluation presented in this section will include all four previously defined components in order to authenticate the users. The sole purpose of integrating various authentication components is to eliminate the errors of individual authentication methods by combining multiple authentication methods. Each previously introduced authentication method introduces a certain amount of error that is represented by the accuracy of the particular approach. Given that the error of each individual method is occurring at a different point, this provides the ability to the method that correctly infers the authentication result, to compensate the error for the components that could not estimate correctly the ID of the user. The following subsections provide a more detailed analysis of the evaluation performed

The overall system is based on a voting mechanism that fuses the different machine-learning based authentication components that are based on social interaction, location, distance and speed patterns. Given that the different approaches managed different accuracy, it was considered to add different weights on each of the authentication processes. In particular the weights assigned to each of the components are derived from the accuracy of the evaluation described in the aforementioned sections. The weights assigned at each component are:

- Social interaction patterns weight: 0.79
- Location patterns weight: 0.89
- Distance patterns weight: 0.92
- Speed patterns weight: 1.5

It could be noted that the majority of the components have been assigned the accuracy score as a weight except from the speed patterns weight which has a weight of 1.5. The threshold for classifying user as owner is over 1 and below 1 is imposter. This is because the speed patterns component managed very high accuracy for authenticating people, which was 0.98. In order keep the high score, the speed patterns component was considered as a dominant factor in the authentication process. By providing 1.5 weight, even in the case where rest of

the authenticators will misclassify, the accuracy of the speed pattern will dominate the result and discard the result of the other classifiers.

3.3.5.1 Accuracy

Similarly to the previous sections, the overall system was evaluated initially in terms of accuracy. A measure of the approaches error is provided by understanding and quantifying the performance of the system in terms of accuracy. The evaluation was performed on the same dataset provided by StudentLife. The system considers all four behavioural cues in the inference process while 10-fold cross validation was utilised in order to quantify the accuracy of the overall system.

The overall system managed to outperform the performance each of the above individual machine-learning models by achieving 0.988294 accuracy. At the beginning of the system design and the evaluation of the system, an assumption was made regarding the error compensation due to multiple different authentication approaches. In particular, given that the error of the different authentication models will occur at different instances, then the various authentication mechanism would compensate and eliminate the error of the each other. This constitutes an indication that the initial assumption made, as the overall accuracy of the system increased with respect to each of the individual authenticator models. In essence, the overall system managed to compensate 0.45% of the error of the most accurate authentication model which was the one that takes into account the average speed of the used in the environment.

3.3.5.2 Confusion matrix

Following, a more detailed evaluation of the overall system was performed in order to understand where the errors are occurring and what is the error that is occurring per user. For this evaluation, the same dataset was used. The dataset was split into training and testing set. The training set was 80% of the initial dataset and the testing set was 20% of the initial dataset. The results of this evaluation are presented in Table 3.15 through a confusion matrix.

Table 3.15. Confusion matrix for the decision tree model.

ID	0	1	2	3	4	5	7	8	9	10
0	**1429**	0	5	1	2	2	0	0	0	2
1	0	**1314**	7	2	0	2	1	12	8	1
2	1	7	**3713**	0	10	17	6	9	7	11
3	1	0	0	**538**	0	0	0	0	1	1
4	3	1	9	0	**1448**	12	4	13	6	7
5	1	0	9	0	8	**11778**	8	2	9	6
7	1	3	2	0	1	1	**1814**	4	1	1
8	0	4	5	0	7	3	3	**3567**	3	3
9	0	7	4	0	3	6	2	3	**2776**	4
10	0	0	6	1	4	10	4	4	10	**3185**

Table 3.15 provides the results of the evaluation through a confusion matrix, indicating the errors occurring among the different users. The best performance is observed at user 5, where at 11821 estimations the minimum error was 0 and the maximum error was 9 faulty estimations. Following, users 0, 3, 7, 8, 9 and 10 achieved also the perfect minimum error and maximum was 5, 1, 4, 7, 7 and 10 faulty estimations. One would consider user 3 as the best user to authenticate given that the maximum faulty estimation per user is 1, however the number of samples used for the evaluation is smaller (541) than for the other users. Next, users 1 and 2 managed also in some cases to achieve perfect performance per user with 0

faulty estimations and regarding the maximum number of faulty estimations they performed 12 and 17 respectively for 1347 and 3781 number of samples. Lastly, user 4 achieved with respect to some users the perfect performance, and for the maximum number of users it performed 13 faulty estimations for 1503 number of samples. It should be noted that the overall system managed to estimate in best case for every user without performing any mistake and in worst cast 17 erroneous estimations were performed.

3.3.5.3 Other performance metrics

Following to the evaluation through the confusion matrix, additional performance metrics were leveraged in order to provide in depth analysis of the performance of the overall system. The confusion matrix provided an indication of the erroneous estimations, among which users these are taking place, what is the minimum number of erroneous estimations per user and what is the maximum number of erroneous estimations that were managed through the evaluation. The other performance metrics such as precision, recall, f1-score and support were used to evaluate the robustness of the overall BehavAuth system in authenticating multiple users. For the evaluation process the dataset was split again 80% as training set and 20% as testing set. The results of the evaluation are presented in Table 3.16 below.

Table 3.16. Other performance metrics for the decision tree model.

Class	Precision	Recall	F1-score	Support
0	1.00	0.99	0.99	1441
1	0.98	0.98	0.98	1347
2	0.99	0.98	0.98	3781
3	0.99	0.99	0.99	541
4	0.98	0.96	0.97	1503
5	1.00	1.00	1.00	11821
7	0.98	0.99	0.99	1828
8	0.99	0.99	0.99	3595
9	0.98	0.99	0.99	2805
10	0.99	0.99	0.99	3224
Avg / Total	0.99	0.99	0.99	31886

Table 3.16 provides the results of the evaluation of the BehavAuth overall system in terms of precision, recall, f1-score and support. The highest performance was achieved by the BehavAuth system at user 5, where the precision, recall and f1-score is on the highest/perfect rate 1.00 for all three metrics and considering 11821 number of samples. User 0 managed also to achieve 1.00 precision with a 0.01 decrease at recall and f1-score. Users 3, 7, 8, 9 and 10 managed 0.99 for the f1-score metric and the recall metric. Users 7 and 9 managed 0.98 recall and users 3 and 8 managed 0.99. Following, users 1 and 2 achieved 0.98 f1-score and recall while the precision was 0.98 for user 1 and 0.99 for user 2. The f1-score for user 4 was 0.97, the recall 0.96 and the precision was 0.98. In overall, BehavAuth performed very well in terms of all metrics, where precision was between 0.98-1.00, recall was between 0.96-1.00 and f1-score was between 0.97-1.00.

Section 4 - User experience evaluation

This section is focused in evaluating in the user experience in terms of different parameters such as the delay in authentication, delay in the training process and the power consumption.

4.1 Delay in the training process

This subsection discusses about the evaluation of the delay in the training process. In order to be able to create the machine-learning models, there is a need to go through a learning process. During the learning process, the mobile app collects sensor data and labels them based on the input of the user (use of user or imposter training button). After the end of the data collection process, the users can select to train the models and allow them to initiate the authentication process.

As discussed in previous section, the BehavAuth app in order to learn the patterns of the users in terms of social interactions, location, speed and distance there is a need to monitor the daily routines of the users for a certain period. Our evaluation showed that the longer the training period is, the higher the classification accuracy, which is justified as the mobile app has a longer period to learn the behavioural patterns of the user.

The proposed time for the training process is at least 1 week. The users need to collect sensor data for 1 week and then train the machine-learning models. Training the machine-learning models for less than a week may lead to decreased accuracy of the authentication process. The 1 week period is able to include the weekly patterns of the users and enable the differentiation among other users.

4.2 Delay in authentication

This subsection is dedicated to the evaluation of the delay in the authentication process. The authentication process is followed by training process that was evaluated in the previous subsection. In the authentication process the BehavAuth app collects sensor data from the Bluetooth and location interfaces and then forward that information to the machine-learning components that perform the authentication of the user.

The execution of the authentication module is highly dependent on the Bluetooth and location interfaces. The mobile app is developed using an event-based architecture. This means that the BehavAuth app performs an authentication only when sensor data have been generated. It does not poll the sensor interfaces every certain polling period, as this constitutes a power-hungry process.

In contrast, BehavAuth waits until the sensor data are generated from the Bluetooth and location interfaces. Then, the sensor data are forwarded to the machine-learning models which perform the classification/authentication process. The actual classification/authentication process delay is dependent on the sensing process. This means that the actual classification process takes a few milliseconds, however the sensing process is taking place every 10 seconds for the location and the Bluetooth interface. In case no Bluetooth device is detected, the authentication process is delayed for 10 seconds until the next Bluetooth discovery. In order to perform an authentication process, there is a need to receive the latest known location and a nearby Bluetooth device.

4.3 Power consumption and CPU load

This section focuses on evaluating the power consumption and the CPU load of the mobile app. BehavAuth performs continuous background data collection and inference, which indicates the importance of understanding the power consumption of the mobile app in order not to endanger the user experience.

4.3.1 Experimental setup

In order to evaluate the power consumption of the BehavAuth mobile app, a particular experimental setup was designed. The evaluation was performed on a Google Nexus 5 mobile phone with commercial ROM Android 6.0.1. QUALCOMM offers a professional mobile app that enables monitoring of different aspect of the mobile phone and the installed mobile apps. The Tredn app was used to profile the BehavAuth app in terms of power consumption and the CPU load. The mobile phone had preinstalled usual apps that a common mobile device could have. The WiFi interface was connected and cellular network was active. Also, the screen of the mobile phone was turned on during the whole duration of the experiment.

The experiment was divided into three sections:

- **Default**. In this case, the mobile app was running the default app of a common mobile device. WiFi and cellular were enabled.
- **Only sensors**. This configuration is the same as the **Default**, but additionally the Bluetooth and the Location sensors were enabled.
- **BehavAuth app**. This configuration is the same as the **Only Sensors**, but additionally the BehavAuth was running and performing authentication of the user.

The mobile phone was kept on the same screen backlight setting during all three experiments and the rest of the running mobile apps were the same in all three cases. The only changes that were applied were the aforementioned.

The three different experiments were introduced for the user to understand the effect of using only the Bluetooth and Location sensors as well as the effect of the BehavAuth app in the power consumption and the CPU load.

4.3.2 Performance metrics

To evaluate the power consumption and the CPU load the following performance metrics were defined. Initially the average power consumption and the average CPU load were calculated for all three experiments. This would provide an initial indication of the effect of using the Bluetooth and Location sensors as well as of the BehavAuth app. The power consumption is calculated in uW and the CPU load in percentage. The Nexus 5 has internally 4 different CPUs and other GPUs. In the CPU load, the overall CPU usage was considered through a merged number that was provided in percentages.

4.3.3 Results

This section provides the results of the evaluation of the BehavAuth app in terms of power consumption and CPU load.

4.3.3.1 Power consumption

This subsection provides the results of the power consumption evaluation through the three different experiments that were described in the above experimental setup section. Table 4.1

provides the results of the evaluation of the power consumption in the three experiment setups a) default, b) only sensors and c) the BehavAuth app.

Table 4.1. Average power consumption in the three experiments.

Default	Only sensors (uW)	BehavAuth (uW)
359603.68	415525.184	426022.412

The average power consumption of the commercial off-the-shelf smartphone device was used as a summarising measure of the performance of the power consumption of the smartphone. In the default case, the smartphone does not have installed the BehavAuth app and also the Bluetooth and location sensors are turned off. The expected result of the default experiment is that it will achieve the lower power consumption of each of the three. Table 4.1 validates this assumption by providing an average power consumption of 0.359W. In the second experiment, the Bluetooth and location sensors were activated. As expected an increase in the power consumption was observed, around 0.056W. Then after having activate the Bluetooth and location interfaces the BehavAuth app was installed and executed. The training process was not considered during this experimentation, only the classification/authentication process. In the third experiment the BehavAuth was performing classification/authentication and as expected there was an increase of the power consumption of around 0.011W. Thus, with respect to the default operating mode of the smartphone, the use of only sensors showed an increase of 15.6% in the power consumption and the use of the BehavAuth app showed an increase of 18.5%. At this point the mobile app does not perform any optimisations regarding the data collection process except from the overall event-based architecture. However, the authentication process based on the machine-learning models provided only a 2.9% increase in the power consumption.

4.3.3.2 CPU load

This subsection provides the results of the CPU load evaluation through the three different experiments that were described in the above experimental setup section. Table 4.2 provides the results of the evaluation of the CPU load in the three experiment setups a) default, b) only sensors and c) the BehavAuth app.

Table 4.2. Average CPU load in the three experiments.

Default (%)	Only sensors (%)	BehavAuth (%)
19.4511312913652	19.8516465863453	24.9847991313789

The CPU load evaluation was conducted in order to understand if the increase in the computation that is introduced by the machine-learning models is endangering the user experience and if this will affect the usage of other mobile apps. The experimental setup for the evaluation of the CPU load is similar to the one observed in the power consumption evaluation. According to the background of the team, what is expected during this experiment is to receive the lowest CPU load at the default experiment, then the only sensors experiments and finally the BehavAuth will provide the highest CPU load given the machine-learning computations that are conducted.

Table 4.2 provides the results of the three experiments. As observed, the lowest CPU usage 19.5% was logged when the smartphone had the default configuration and none of the Bluetooth/location interfaces was active as well as the BehavAuth app was not installed. Next,

when the Bluetooth and Location interfaces were active a small increase in the CPU load was observed to 19.9%. Then once the BehavAuth mobile app is installed and started performing authentication, the CPU usage increased to almost 25%. The machine-learning models that are executed on the smartphone forced the CPU load to increase by 5.5% from the default mode and 5.1% from the only sensors mode. The percentage of increase constitutes reasonable even for a continuous sensing and inference application, as it does not endanger the user experience and users can continue to use their smartphones in their daily lives without getting affect. It should be mentioned that the smartphone used is 5 years old mobile phone, at the time this document was written. As the smartphones become more powerful year-by-year the CPU load will decrease even more.

Section 5 - Conclusion

This deliverable described the trials and validation activities that were introduced throughout the duration of the BehavAuth project. The validation activities had their main focus towards evaluation of the BehavAuth platform, and provided feedback to the design of the overall system.

This document started by describing an overview of the trials and validation activities performed during the BehavAuth project and further after the end of the project. Initially an overview of the evaluation activities performed was outlined defining the key points of the dissemination activities. Following, the impact achieved through the BehavAuth project and the foreseen trials during and after the end of the project as Quadible will ensure its continuation.

Next step was to evaluate the social interaction detection mechanism that was developed for the BehavAuth project, using real-world experiments in indoor environments. The evaluation was performed using off-the-shelf mobile phones that had deployed a data collection and social interaction detection mobile app. The evaluation was performed in different indoor environment i.e. two meeting rooms and one corridor of an office environment. The evaluation showed in most cases that the social interaction detection mechanism was able to detect on-going social interactions based only on the interpersonal distance of the users. In some cases where the users were very close to each other the approach had downgraded accuracy in terms of detecting when the users were in close proximity and did not interact with each other. In overall the accuracy of the social interaction detection mechanism given the trade off with the computational complexity of introducing a relative orientation mechanism was considered sufficient.

Following, the trial and validation report continued through the evaluation of the different authentication components developed throughout the BehavAuth project. The section provided a detailed description of the dataset used in the evaluation process, the performance metrics used to benchmark the different components and finally the results of the evaluated models are presented and discussed. In particular, the dataset included an explanation of the data collection process, the pre-processing performed and finally a discussion of the class distribution. Then the performance metrics used in the evaluation process were discussed including the accuracy of the different algorithms, the confusion matrices to understand the error occurrence and then the robustness measure through precision, recall and f1-score. Having defined the dataset and the performance metrics, the section discussed about the evaluation of the different authentication components based on the previously defined performance metrics. In particular, the evaluated performed validation of the authentication mechanism components based on the social interaction detection, location patterns, speed and distance patterns in the environment. The highest performance was observed in the speed and distance patterns of the users indicating their significance as behavioural cues in the authentication process.

The evaluation of the BehavAuth system indicated the highest accuracy among the various authentication components, achieving 98.8% accuracy. All four authentication modules were used in the system including social interaction, location, distance and speed patterns. The high accuracy of the distance and speed patterns, enabled mainly the system to achieve such a high accuracy. However, the other authentication components with lower accuracy were still considered to improve the robustness of the system and to cope with some of the error of the speed and distance patterns, which was proved by increasing the accuracy by 0.45%.

Next, the BehavAuth mobile app was evaluated in terms of user experience. In particular, different parameters were considered. For the delay in the training process, there is a need of at least 1 week to train the machine-learning models and to learn the weekly behavioural patterns of the users. For the delay in the authentication process, the authentication period is highly dependent on the existence of nearby device in order to allow the social interaction detection mechanism to infer. Also, the BehavAuth app was evaluated in terms of the power consumption of the app as well as the Bluetooth and Location interfaces, which indicated an increase of the power consumption because of the BehavAuth app of 2.9% with respect to the idle usage of these sensors. Then, the CPU load also was evaluated in order to understand if the machine-learning models, endanger the user experience. The evaluation showed that an increase of only 5.1% was observed because of the BehavAuth app, which is likely to decrease in newer smartphones that have more powerful processors.

This document constitutes a report about the trials and validation activities performed throughout the BehavAuth project in order to evaluate the performance of the developed work done in both the BehavAuth and TagItSmart project.

Section 6 - References

1. BehavAuth D1.2 – Final system design based on feedback from early trials
2. BehavAuth D1.1 Initial design based on the use case and requirement analysis
3. TagItSmart EU project – www.tagitsmart.eu
4. Wang, Rui, Fanglin Chen, Zhenyu Chen, Tianxing Li, Gabriella Harari, Stefanie Tignor, Xia Zhou, Dror Ben-Zeev, and Andrew T. Campbell. "StudentLife: Assessing Mental Health, Academic Performance and Behavioral Trends of College Students using Smartphones." In Proceedings of the ACM Conference on Ubiquitous Computing. 2014.
5. Inman, James *(1835) [1821].* Navigation and Nautical Astronomy: For the Use of British Seamen *(3 ed.). London, UK: W. Woodward, C. & J. Rivington*. Retrieved 2015-11-09.
6. Haversine formula error - https://geopy.readthedocs.io/en/1.10.0/
7. Vincenty T. Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. Survey review. 1975 Apr 1;23(176):88-93.
8. *Fawcett, Tom (2006).* "An Introduction to ROC Analysis" *(PDF). Pattern Recognition Letters*. 27 (8): 861–874. doi:10.1016/j.patrec.2005.10.010.
9. Powers, David M W (2011). "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation" (PDF). Journal of Machine Learning Technologies. 2 (1): 37–63.
10. Hall, E.T., 1966. The hidden dimension.
11. Seber, G.A. and Lee, A.J., 2012. Linear regression analysis (Vol. 329). John Wiley & Sons.
12. Mika, S., Ratsch, G., Weston, J., Scholkopf, B. and Mullers, K.R., 1999, August. Fisher discriminant analysis with kernels. In Neural networks for signal processing IX, 1999. Proceedings of the 1999 IEEE signal processing society workshop. (pp. 41-48). Ieee.
13. Peterson, L.E., 2009. K-nearest neighbor. Scholarpedia, 4(2), p.1883.
14. Safavian, S.R. and Landgrebe, D., 1991. A survey of decision tree classifier methodology. IEEE transactions on systems, man, and cybernetics, 21(3), pp.660-674.
15. John, G.H. and Langley, P., 1995, August. Estimating continuous distributions in Bayesian classifiers. In Proceedings of the Eleventh conference on Uncertainty in artificial intelligence (pp. 338-345). Morgan Kaufmann Publishers Inc..