Eye tracking-based Sedentary Activity Recognition with Conventional Machine Learning Algorithms

Violeta Chatalbasheva
Supervisor(s): Guohao Lan, Lingyu Du
EEMCS, Delft University of Technology, The Netherlands

June 21, 2022

A Dissertation Submitted to EEMCS faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering

Abstract

Sedentary activity recognition is an important research field due to its various positive implications in people's life. This study builds upon previous research which is based on low level features extracted from the gaze signals using a fixation filter and uses a dataset of 24 participants performing 8 different sedentary activities. The main research question are related to extracting features from the raw data and selecting the most relevant ones which improve the classification accuracy. The novelty of this paper is using dynamic thresholds in the fixation filter to ensure the fixation-specific measurements reported by literature as well as contributing to the human activity recognition (HAR) field by developing an additional low-level gaze feature in combination with the fixation dispersion area. The machine learning (ML) models, Random Forest, k-NN (k-Nearest Neighbor) and SVM (Support Vector Machine), used for the classification task are evaluated using the within-dataset evaluation protocol, with cross validation and hyperparameter tuning. The overall recognition accuracy of the Random Forest model is 0.94 (f1-score).

1 Introduction

1.1 Background

The past decades have seen the rapid development of tracking eye movements to classify different activities and cognitive tasks [1, 2]. Analyzing gaze signals has various benefits. For example, Kunze's work presents a software system counting the words read by a person which works as a tool for cognitive fitness [3]. This proves that eye-tracking can be used to monitor what people read and how much they understand in order to improve their learning ability, language and critical-thinking skills [1, 3]. Another positive aspect of eye tracking is improving the system's response depending on the user activity. For instance, in 2006 Toyota has introduced an advanced system, monitoring the driver's activity which processes the gaze direction data to prevent car drivers from being distracted [4].

1.2 Related work

Researchers have taken a stride towards recognizing sedentary activities from the gaze signals of eye tracking devices using machine learning (ML) techniques. Most papers in this field use conventional ML classifiers [2, 5, 6] rather than deep neural networks (DNNs). The reason for that is the limitation in the size of the available datasets. To guarantee accuracy and efficiency, DNNs require a large dataset, which is technically infeasible due to the numerous gaze-sensing conditions (e.g. various lighting conditions) [7]. This is one of the common challenges in the domain of recognizing sedentary activities using data obtained from eye tracking. Therefore, the focus of the research in this field is centered around conventional ML methods [2, 6] and so is the current conducted research.

However, there are other challenges which come with the use of ML methods - feature extraction and feature selection. Feature extraction is fundamental for achieving good classification accuracy as it is what the ML model is trained on to perform well the classification task. Interestingly, the major concern in feature extraction in the eye tracking-based activity recognition domain stems from the heterogeneity in human visual behaviour. Since no two people will view the same content in the same way, it is very hard to generalize the extracted features in a wider range of use cases, such as different subjects [7] and visual stimuli [6]. This significantly hinders the development of robust classifiers for this task. Feature selection aims to maximize the variability in the data in as small subset of features as possible. This has a positive impact on both the classifier's training time and the performance. The main issue here is determining the best features which result in good performance accuracy as existing research does not show a common procedure to answer this question. These challenges have not yet been solved completely which is why research in this area has been generating a lot of interest. Thus, there is still a lot of room for such experiments and revolutionary findings.

1.3 Research questions and main contributions

Therefore, the aim of this research is to investigate the use of conventional ML algorithms to recognize different sedentary activities (e.g. read, browse, watch) from the gaze signals captured by an eye tracker. It focuses on extracting low-level gaze features to implement and evaluate three conventional ML classifiers: Random Forest, SVM and k-NN. The two main research questions (RQ) this paper answers are directly linked to the challenges of feature extraction and feature selection: **how to design and implement different feature extraction methods for eye movement signals** and **what the best features for training conventional ML algorithms are to achieve good recognition accuracy**. The 3rd RQ is about the **impact of different subjects on the recognition performance**. This can improve the generalizability of ML models in the research field. Finally, inline with the 4th RQ, this research **compares the above algorithms on three different datasets in terms of accuracy, memory usage and robustness against heterogeneity among subjects**.

In summary, the key contributions in this study are: (a) developing dynamic thresholds for the fixation filter to guarantee the accuracy in the fixation durations, mean and standard deviation known from the literature (3.2) and (b) the development of a new fixation-based feature called fixation-radius (2.2.2).

The outline of this research paper is the following. Chapter 2 dives into the details of the methodology discussing the domain-specific challenges with respect to each stage of the ML classification pipeline. The performed experiments and results are described in chapter 3, followed by discussion of the ethical implications and the reproducibility of the results in chapter 4. Finally, chapter 5 summarizes the results and gives an overview of some improvements which can be made in the future.

2 Methodology

Classification tasks in ML are solved by using the pipeline: data preprocessing (2.1), feature extraction (2.2), feature selection (2.3), training the model and model evaluation (2.4), illustrated in Figure 1. Each of the subsequent sections dives deeper into the actual steps required to perform the activity recognition task.

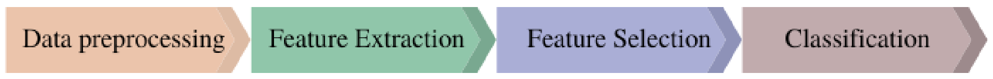


Figure 1: An illustration of the ML working pipeline.

2.1 Data Preprocessing

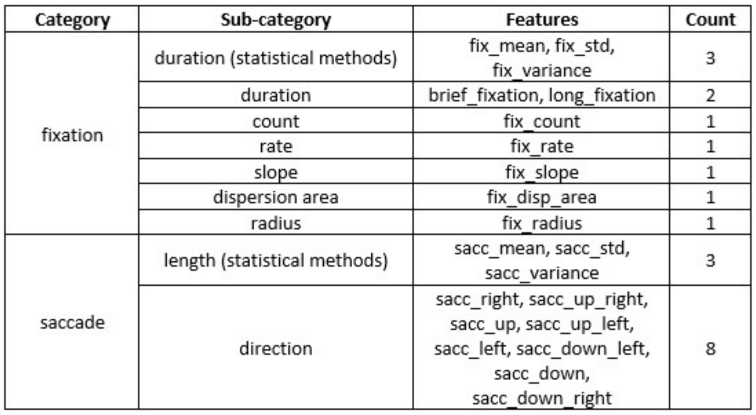
This step aims to prepare the data for the classification task: filter the outliers to clean the data and ensure its quality for achieving good performance accuracy. Particularly, a median filter with a sliding window of 500ms has been applied to remove the outliers as suggested in [5, 7]. The filter removes values which have a large Euclidean distance to the other points in the sliding window. The benefit of a small sliding window is that it observes the local changes in the gaze signal to apply the filtering without reducing the variability in the data. The value of 500ms was determined empirically by comparing the classification accuracy to see how this parameter influences the results. Then, the data, consisting of gaze coordinates: x and y values, is normalized between 0 and 1 to eventually improve the classification accuracy. The normalization formula shown in Figure 12, Appendix A uses the minimum and maximum value computed from the coordinate arrays. However, the size of the screen used in the data collection process is reported as 1920x1080 which means that the x and y coordinates range between 0 and 1920 and between 0 and 1080, respectively. Consequently, the maximum values are fixed at 1920 and 1080 for the x and y coordinate arrays, respectively. Then, the data is rescaled to fit in the desired range.

2.2 Feature Extraction

The goal of the feature extraction is to transform the raw data into a set of features that capture the activity-related information from the data to train the classifiers. But first, the raw data is transformed into fixations and saccades - the two fundamental eye movements [7].

Existing research [5, 6] has focused on using low-level gaze features: "statistical measures derived from the eye movement properties" [2]. They can be separated into two main groups: fixation- and saccade-based features. A fixation is the process of maintaining the focus of one's gaze at a specific point, whereas a saccade is the rapid eye movement between two consecutive fixations. In their study, the authors of [2] have combined low- and mid-level[1] gaze features to see if that could increase the classification accuracy. However, after their thorough examination, the results have not led to any significant improvements: an increase of up to 4 %. Therefore, this research focuses on low-level gaze features.

This research makes use of 21 low-level gaze features summarized in Table 1.



Category	Sub-category	Features	Count
fixation	duration (statistical methods)	fix_mean, fix_std, fix_variance	3
	duration	brief_fixation, long_fixation	2
	count	fix_count	1
	rate	fix_rate	1
	slope	fix_slope	1
	dispersion area	fix_disp_area	1
	radius	fix_radius	1
saccade	length (statistical methods)	sacc_mean, sacc_std, sacc_variance	3
	direction	sacc_right, sacc_up_right, sacc_up, sacc_up_left, sacc_left, sacc_down_left, sacc_down, sacc_down_right	8

Table 1: An overview of all fixation-based and saccade-based features.

2.2.1 Fixation Filter

To start the feature extraction process, the raw data needs to be transformed into fixations and saccades. The fixation filter, described in Olsson's work [8], has turned out to be the most common approach applied in the domain of eye tracking-based sedentary activity recognition for extracting fixations [6, 7]. The code of the fixation filter is shown in code snippet 1.

[1]An abstraction layer over the low-level gaze features better representing the activities [2]

```python

def fixation_filter(x, y):
    # Calculate the differences in means in two sliding windows.
    mean_differences = calculate_mean_differences(x, y, sliding_window_size)

    # Find the peaks in the mean differences representing the saccades.
    peaks = find_peaks(mean_differences)

    # Filter the peaks.
    filtered_peaks = remove_noise_peaks(peaks, sliding_window_size)

    # Find peak indices.
    peak_indices = apply_peak_threshold(filtered_peaks, peak_threshold)

    # Calculate fixations and their durations.
    fixation_positions, peak_indices = estimate_fixations(x, y, peak_indices,
    threshold_radius)

    return fixation_positions, peak_indices
```

Listing 1: Fixation filter.

The fixation filter uses the concept of a moving average filter as it calculates the differences in the means of two sliding windows over the whole gaze signal. The two most important assumptions it makes are based on the changes in the mean differences: if there is a slight change, then a fixation has been detected, whereas if the change is abrupt, then it is classified as a saccade. Once the mean differences are calculated, the algorithm goes on to finding the peaks, representing saccades, by going over the mean differences and keeping the ones that are bigger than the previous and the consecutive value. Then follows a noise reduction technique by keeping only the highest value in a sliding window. Afterwards, a threshold is applied over the calculated peaks to discard the invalid saccades. The final step of the algorithm is to estimate the spacial position of fixations. The indices of the remaining peaks indicate which points from the raw data belong to a fixation. The algorithm takes the median of those points to estimate a fixation. Afterwards, a threshold radius is applied to join fixations that are spatially closer together.

2.2.2 Fixation-based features

The first fixation-based features are based on some methods of statistical analysis: mean, standard deviation and variance of the fixations duration. The duration of fixations can be estimated using the number of points forming a fixation divided by the sampling rate (30 Hz) of the eye tracker (Tobii X2-30 eye tracker [9]). Two other features are extracted from the fixation durations - the number of brief fixations (<500 ms) and long fixations (>500 ms). Furthermore, the number of fixations also serves as an indicator of the sedentary activity. The fixation rate and the fixation slope are considered as important features by the author of [2]. The fixation rate represents the number of estimated fixations per second, whereas the fixation slope is calculated using linear regression over the fixations [6]. The features fixation radius and fixation dispersion area [10] represent how close fixations are to each other. They compute the distance between the mean of all fixations and each fixation, sorting these distances and taking a 75% threshold to remove outliers [2]. Then the longest distance within the remaining fixations is picked to represent the radius, while the dispersion area is based on a rectangle which encapsulates these fixations. The fixation radius is a new feature, developed for the purpose of this research in combination with the fixation dispersion area to make a more precise prediction of the area of interest (AoI).

2.2.3 Saccade-based features

The basic saccade-based features are the mean, standard deviation and variance of the saccades' lengths. The length of the saccades is computed by taking the Euclidean distance between two consecutive fixation estimates. Then, the authors of [2, 6] look at the number of saccades in different directions. This research uses 8 directions on every 45°. This is done by finding the angle between two consecutive fixations and increasing the counter of the respective saccade angle sector the computed angle belongs to.

2.3 Feature Selection

The aim of the feature selection step is to keep only the relevant features while removing the redundant ones that capture recurring information. This leads to a reduction in the computational cost and the model training time, as well as an improvement in the classification accuracy and it helps avoid overfitting the data [11]. Two methods are considered for selecting the most relevant features: Univariate selection [12] and mRMR (minimum-Redundancy Maximum-Relevance) [13]. They both use the statistical measure $mutual\ information$ which is specifically suitable for numerical input and categorical output data and measures the dependency relationships between features. The univariate selection looks at each feature individually and how strongly it influences the target variable whereas the mRMR feature selection method not only finds which features have a strong relationship with the target variable but also how they correlate to the previously selected features [14]. So, these methods will be applied to the SVM and k-NN models as the Random Forest model has a built-in feature importance method [12].

2.4 Classification

This research evaluates the performance of three conventional ML classifiers on recognizing different sedentary activities from the SedentaryActivities dataset (3.1). The within-dataset evaluation protocol is used to present the results (3). As its name suggests, the classifiers are built and evaluated on the same dataset which is divided into training and test set at random [15]. The reason behind choosing within-dataset evaluation (over cross-dataset evaluation[2]) is that this study already compares the obtained results with other datasets based on the work of peers (3.3.4). For the purpose of this study, the data is split in the following ways: (a) such that 80% of each subject appears in the training set and the remaining 20% are part of the test set and (b) such that each subject is tested independently once while the classifiers have been trained on the rest of the subjects.

To evaluate the performance of the three conventional ML models, two evaluation schemes have been applied: k-fold cross validation across all participants and leave-one-person-out cross validation (LOOCV). The k-fold cross-validation (k=5) has been performed together with hyperparameter tuning to further improve the classification accuracy while the LOOCV method is used to analyze the ability of the classifiers to predict sedentary activities performed by a new participant.

3 Experimental Setup and Results

This chapter describes the specifics of the dataset used in this research (3.1). Then, it dives deeper into the details of the actual parameter estimation in the fixation filter (3.2). Finally, it discusses the results of the classification (3.3).

[2]Trains the classifiers on one dataset and tests them on another one.

3.1 Dataset

The provided dataset is called SedentaryActivities and consists of 24 subjects performing 8 sedentary activities. The activities are divided in two groups: common desktop activities - read, browse, play, search, watch, and software engineering activities - write, interpret, debug. The data consists of raw gaze points: x and y coordinates, as well as the specific time a data point has been recorded at (in ms.): timestamp. The specific tasks, the participants were asked to perform for the data collection, are varied to introduce greater variability in visual stimuli and to avoid creating a bias in the classification [2]. As remarked by Kunze in [6], ML classifiers might recognize the visual stimulus instead of the activity itself if only one stimulus is used in the data collection process. Therefore, it is very important not to repeat the task per activity (e.g. reading "an excerpt from a book, an article, and a short story" [2]).

3.2 Task-specific Parameter Estimations

A crucial step in this research is estimating fixations and saccades from the raw data. As discussed in section 2.2.1, the fixation filter finds peaks, which represent the saccadic eye movements, and based on that calculates the fixations. The estimated number of fixations this research is using are based on some statistics reported from previous research [16]. Duchowski finds that the mean of fixation durations is around 540ms and a typical fixation duration ranges between "150-600ms and 90% of the viewing time is devoted to fixations" [16]. The fixation filter makes use of three parameters which require tuning.

- Sliding window length: indicates over how many samples a peak will be estimated.
- Peak threshold: identifies true peaks.
- Threshold radius: defines how close to each other fixations are.

A dynamic approach for finding the two thresholds has been developed in this research. It has proven to be the most effective way of estimating these parameters so that the statistics related to fixations, mentioned above, are satisfied and the determined thresholds are activity-specific. Both of those arguments guarantee good classification performance.

3.2.1 Sliding window length

Several experiments have been performed with the length of the sliding window used to estimate the peaks. Looking at a larger window reduces the impact of noise. However, a larger sliding window leads to a greater reduction in the number of fixations which contradicts the statistics reported in the work of Duchowski [16]. This is because the algorithm looks at the large sliding window and keeps only one peak. While, if the sliding window is smaller, the loss of saccadic movements will be prevented. So, to satisfy the findings of Duchowski and preserve the number of fixations, the length of the sliding window in this research is set at 5 samples. This value is very similar to the one used by the author of the algorithm [8], which is 4 samples. Figure 2 on the left shows the major reduction in the number of the estimated fixations with a sliding window of 50 data points compared to the fixations estimated with a sliding window of 5 data point displayed in Figure 3 on the right.

3.2.2 Peak threshold

To find a good threshold for the peaks, a histogram plot is very useful as it shows the frequency of the peaks in different ranges of the peak values. Similar to the length of the sliding window, the value of the threshold influences the number of fixations. The higher the threshold, the fewer fixations. Therefore, the threshold needs to be set at a value such that it will keep the majority of

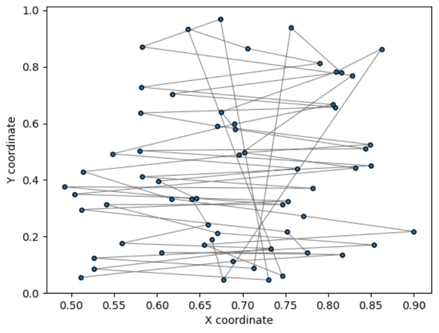


Figure 2: The estimated fixation positions of the reading activity of subject 1 with a sliding window of length 50 samples.

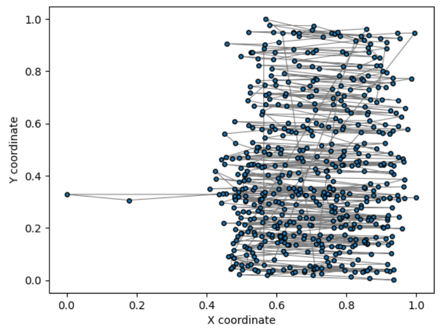


Figure 3: The estimated fixations positions of the reading activity of subject 1 with a sliding window of length 5 samples.

the peaks to guarantee that fixations will not be lost and to still be able to distinguish correctly between true and false saccades. To estimate the threshold, this research uses 90% of the peak values. Figure 4 shows the histogram of the filtered peaks after the noise reduction step.

The current implementation picks the peak value at the 10% position for a threshold. In the case of Figure 4 there are 777 peak values, ranging between 0.0001 and 0.7547 but only a sample of them is shown in the histogram. So, the 77th peak value is used for the value of the threshold which corresponds to 0.0117.

3.2.3 Threshold radius

The function of the threshold radius is to identify spatially closer fixations which essentially represent the same fixation. Therefore, they need to be merged. This, again, impacts the final number of estimated fixations. So it should not be very large as it would merge two different fixations. The same reasoning as for the peak threshold is applied (3.2.2).

The fixation filter algorithm works by recalculating the distances between the estimated fixations and checking if the shortest distance is smaller than the threshold radius. If so, it removes the peak index that separates the two fixations. Code snippet 2 in Appendix B illustrates the estimation of fixations positions. This research uses the initial estimation of fixation positions to determine the appropriate threshold radius for the activity. It takes the distance at 10% of the sorted distances between consecutive fixations as a representative threshold for spatially close fixations. In the example of Figure 5 the distances between fixations are 698, ranging between 0.01 and 0.99, and only a sample of them is shown in the histogram plot. The estimated threshold radius corresponds to the 69th distance value which equals 0.0316. This results in 637 fixation estimates for the browse activity of subject 1.

3.3 Results

The aim of this study is to find which features contribute to the accuracy of the SVM, k-NN, Random Forest classifiers the most (2nd RQ). Section 3.3.1 compares the performance of the three ML models across all participants using different window sizes for feature extraction (3.3.1). Then, section 3.3.2 compares the performance of the three classifiers after applying feature selection techniques and discusses the impact of the fixation- and saccade-based features on the recognition accuracy. Section 3.3.3 shows the performance accuracy per subject and section 3.3.4 compares the results of this research with the results of peers.

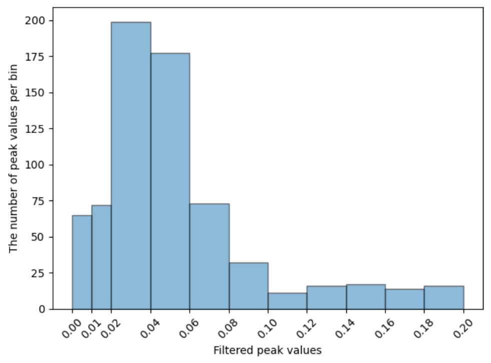


Figure 4: The filtered peaks of the browsing activity of subject 1.

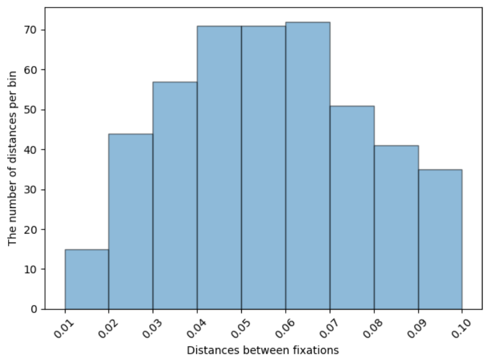


Figure 5: The initial fixation positions of the browsing activity of subject 1.

3.3.1 Person-Dependent analysis

This analysis uses 80% training data per participant and tests the classifiers on the remaining 20% of each participant, i.e. person-dependent analysis. When training, 5-fold cross validation has been performed to tune the hyperparameters, using a grid search [17]. The best hyperparameters are summarized in Table 2.

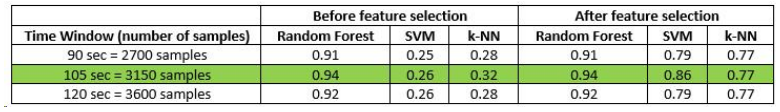
Ransom Forest	SVM	k-NN
n_estimators=1100	C=10, gamma=0.001, kernel="rbf"	k=13

Table 2: Table of the best performing hyperparameters.

The hyperparameter of the Random Forest classifier $n\_estimators$ is tuned at 1100 trees. The SVM model uses a "rbf" kernel to separate the data with a non-linear hyperplane to account for the heterogeneity in human visual behaviour. The C parameter regulates the number of misclassified observations while the $gamma$ parameter represents how exact the fitting of the training data will be [12]. Therefore, to avoid overfitting, neither of those parameters should be a high value: C=10, $gamma$=0.001. The smaller the k of the k-NN classifier is, the higher the chance of the model overfitting on the dataset: k=13. All of these hyperparameters comply with [2].

Performance based on window length

Previous research has shown that building robust classifiers requires features to be extracted from time windows of the gaze signals instead of once for the whole signal [5, 6, 18]. The reason behind this is generating more data for training the classifiers which can avoid overfitting and increase the accuracy and generalizability of the results. This research studies the performance of the classifiers using features extracted on windows based on the analysis in [2]. A summary of the results can be seen in Table 3.



	Before feature selection			After feature selection		
Time Window (number of samples)	Random Forest	SVM	k-NN	Random Forest	SVM	k-NN
90 sec = 2700 samples	0.91	0.25	0.28	0.91	0.79	0.77
105 sec = 3150 samples	0.94	0.26	0.32	0.94	0.86	0.77
120 sec = 3600 samples	0.92	0.26	0.28	0.92	0.79	0.77

Table 3: The f1-scores of the each of the ML models used in this research with respect to the window size used for extracting the features.

This table compares the f1-score accuracy of all activities per classifier before and after feature selection (3.3.2) with respect to different time window sizes. The data in two consecutive sliding windows shares 90% overlap. As the work of Srivastava, Newn and Velloso [2] states, the best performance of the classifiers is found when the features are extracted from a window size of 105s. This holds before and after the feature selection. Overall, it can be seen that the k-NN and the SVM models perform poorly without selecting the most important features. This is because some of the features encode similar characteristics of the data resulting in redundant information. This decreases the number of accurate predictions dramatically. One disadvantage of the k-NN classifier related to this classification task is that it stores the training data and does not build an internal model to predict the unseen data points. Then, it simply performs a majority voting between the k nearest neighbours [12]. The heterogeneity in human visual behaviour together with the large number of extracted features hinder the performance of k-NN and it has the lowest results even after feature selection. Therefore, it can be concluded that this model is not a suitable choice for the sedentary activities recognition task using eye tracking data.

Performance per activity

As Table 3 has proven, the Random Forest performs most reliably. Therefore, the analysis of the performance per activity is considered only for this ML model. The chosen evaluation metrics are confusion matrix and f1-score. The confusion matrix displays the number of correctly or incorrectly classified activities with respect to actual and the predicted label. The f1-score combines precision and recall by taking their harmonic mean. Precision represents the number of positive predictions that are correct, while recall estimates the number of actual positives identified correctly. Figure 13 in Appendix C illustrates these formulas. The f1-score is the metric chosen for the performance by activity analysis as it accounts for the distribution of the data and looks at the correctly classified observations only from the positive class which helps avoid data imbalance and penalizes FN more compared to accuracy.

Figures 6 and 7 plot the confusion matrices of the Random Forest and SVM classifiers, respectively, with features extracted from a window size of 105s. Figure 6 shows that 7 out of the 8 activities have excellent rankings, above 0.9. The *read* and *interpret* activities are predicted with the highest f1-score, followed by the *watch* activity, 0.96. This trend continues in Figure 7. The SVM model is also most certain about the *read*, *interpret* and *watch* activities, f1-scores of 1.0, 0.89 and 0.88, respectively. The lowest score of the Random Forest model corresponds to one of the software-related activities - *write*, while the *browse* activity performs the worst according to the SVM classifier. Random Forest has misclassified the *write* activity as the other two software activities *debug* and *interpret* and SVM mislabels the *search* as *browse*, due to the similar nature in these activity pairs.

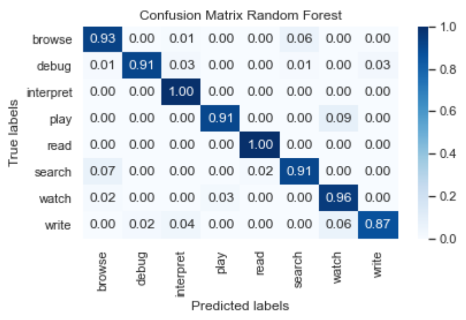


Figure 6: The confusion matrix of Random Forest.

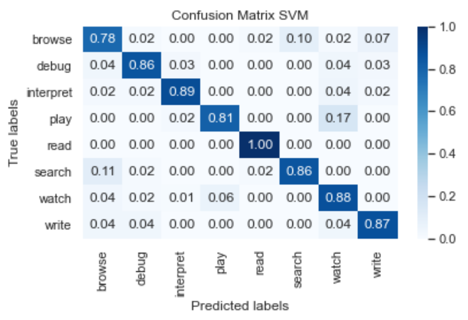


Figure 7: The confusion matrix of SVM.

3.3.2 Feature importance

To find the most important features for good classification accuracy, the classifiers performance must be compared with respect to all features (1) against only the selected ones.

SVM and k-NN

Figures 8 and 9 depict the trend of increasing mean f1-score, shown as a green triangle, with the number of selected features by mRMR for the SVM and k-NN models, respectively. Again, the time window resulting in the best performance of both classifiers, 105s (Table 3), is used.

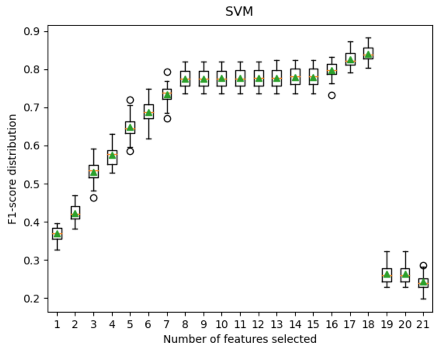


Figure 8: Box-and-whiskers plot for each number of selected features for SVM using mRMR.

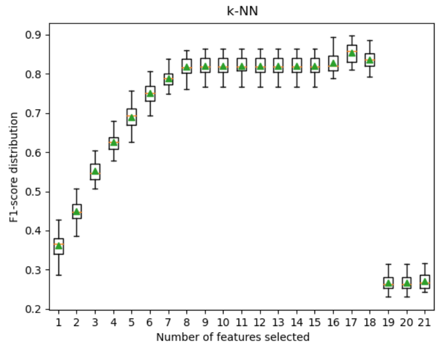


Figure 9: Box-and-whiskers plot for each number of selected features for k-NN using mRMR.

To estimate the optimal number of features that lead to the best classification accuracy, the two models have been trained on the whole dataset and 10-fold cross validated, repeated 3 times for each combination of features[3]. Both figures (8 and 9) aim to show how combinations of different number of features affect the overall level of feature relevance. It can be seen that in the beginning adding more features results in a rapid increase of each classifier's f1-score: until the combination of 8 features. Then, there is a gradual rise in the f1-score with the next 7 added features. Subsequently, there are 3 more features which lead to the highest f1-accuracy score of the SVM model and 2 more of the k-NN classifier, resulting in 18 and 17 selected features for the SVM and k-NN models, respectively. The names of these features are listed in Table 4. Afterwards, the f1-score in Figures 8 and 9 drops significantly. This means that the rest of the features do not add any relevant information to the classifiers capability of predicting the sedentary activities correctly. Furthermore, these plots explain the low results of the SVM and k-NN classifiers, presented in Table 3, when trained on all features.

After establishing the most important features, using the data split 80% training and 20% testing data, the SVM and k-NN models reach their maximum f1-score: 0.86 and 0.77, respectively, on the test data for a window size of 105s. The mRMR[4] feature selection methods was compared to univariate selection[5] as mentioned in section 2.4 but the focus remains on mRMR as it outperforms univariate selection due to considering the relationship between features as well as how each feature influences the result - 0.86, 0.77 (mRMR) vs 0.78, 0.68 (univariate selection), f1-scores of SVM and k-NN, respectively.

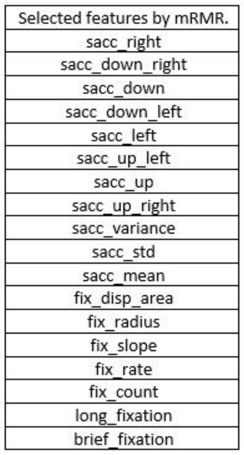
Random Forest

Figure 10 shows the features the Random Forest considers the most important with their rele-

[3]These models are only used to explore the most advantageous combination of features for the final classification task, they are not the ones used to report the final accuracy scores.

[4]https://github.com/smazzanti/mrmr

[5]https://scikit-learn.org/stable/modules/feature_selection.html



Selected features by mRMR.
sacc_right
sacc_down_right
sacc_down
sacc_down_left
sacc_left
sacc_up_left
sacc_up
sacc_up_right
sacc_variance
sacc_std
sacc_mean
fix_disp_area
fix_radius
fix_slope
fix_rate
fix_count
long_fixation
brief_fixation

Table 4: The 18 most important features selected by mRMR for the SVM model. The first 17 features are optimal for the k-NN classification accuracy.

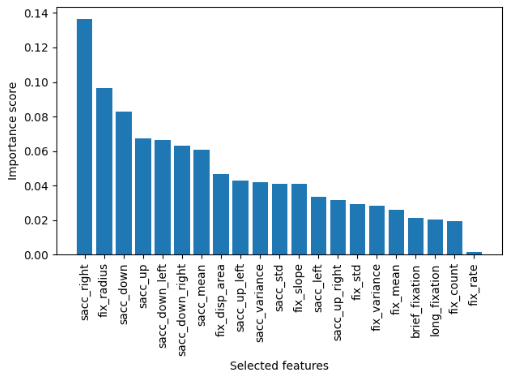


Figure 10: Feature importance scores of the Random Forest classifier.

vance scores. The f1-score performance of the Random Forest model trained on all features and on a selected subset of features stays constant.

From the selected features for each classifier it can be concluded that the features influencing the recognition performance the most are all saccade-based features together with three of the fixation-based features: fixation radius, fixation dispersion area and fixation slope. This can be explained with the fact that fixation-based features are not very indicative of the activity. This is because when the features are extracted from time windows, the fixation-based ones such as fixation count and the fixation duration-based become very similar over the windows as they lose the variability present if they were to be extracted from the whole signal. The expected range of fixation durations [150ms-600ms], as described in [16], gets balanced out and becomes close to [200ms-400ms] over the windows. Whereas, saccade-based features help the ML models much more reliably distinguish between the activities due to several reasons. They show when the subject is consistent in his/her viewing behaviour, meaning certain patterns can be found for some activities (e.g. reading from left to right means that the sacc_'right and sacc_'down_'right have higher importance) and when the behaviour is inconsistent, which is typical for other activities (e.g browsing usually does not have a specific goal so the saccade-based features in all directions will have an impact on the recognition accuracy).

Although not very representative in general, three of the fixation based features (dispersion area, radius and slope) provide some insights into where the subject is looking at. Figure 10 shows that the novel fixation radius feature makes a promising contribution to the relevant features according to the Random Forest classifier. Although, less important for the other two classifiers, the fixation radius appears in the top of the relevance ranking, as seen in Table 4. So, the combination of fixation radius and fixation dispersion area represents the (AoI), where participants focus their attention, while the fixation slope helps determine any patterns of the current visual behaviour (e.g. higher slopes can help distinguish between consistent and inconsistent behaviour). Figure 11 shows the impact of feature selection on the classification accuracy.

3.3.3 Person-Independent analysis

This study makes use of the leave-one-out evaluation method to compare the performance of the models per subject tested. The idea is to train the classifiers on (n-1) subjects and use

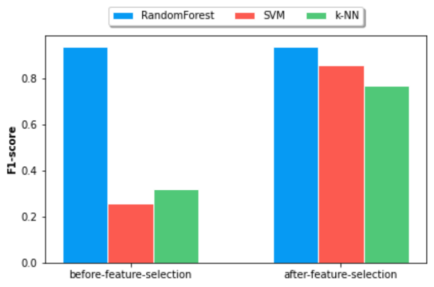
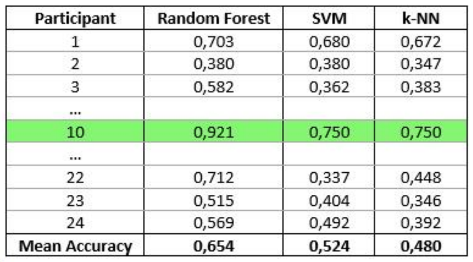


Figure 11: The impact of feature selection on the classifiers recognition accuracy.



Participant	Random Forest	SVM	k-NN
1	0,703	0,680	0,672
2	0,380	0,380	0,347
3	0,582	0,362	0,383
...			
10	0,921	0,750	0,750
...			
22	0,712	0,337	0,448
23	0,515	0,404	0,346
24	0,569	0,492	0,392
Mean Accuracy	**0,654**	**0,524**	**0,480**

Table 5: A summary of the recognition accuracy of LOOCV.

the one that is left entirely for testing, aka. person-independent analysis. This approach helps identify similarities in the visual behaviour of different participants and find how different subjects influence the recognition performance. Table 5 summarizes the classification accuracy per participant and averages them out to obtain the overall accuracy per classifier. The full results can be found in Table 6 in Appendix D.

There is one participant, subject 10, whose visual behaviour is predicted with a very high accuracy by all three classifiers. Overall, the Random Forest scores the best (0.654), followed by SVM (0.524) and k-NN (0.48). The features the classifiers are trained and tested on are extracted from the best performance window size: 105s, but nevertheless, there is a significant drop in performance. This has to do with the amount of people the classifiers are trained on. When a classifier is trained on the visual behaviour of one subject, then it specifically adapts to this person and will very well predict when tested on the same subject. However, when it is trained on multiple subjects it tries to learn many different visual behaviour patterns which makes is harder to predict the activities performed by unseen subjects accurately. This explains the results in Table 5 and also provides some insights into the impact of the heterogeneity of human visual behaviour on the classification accuracy (3rd RQ), namely that it reduces the classification accuracy of the conventional ML models.

3.3.4 Cross-dataset comparison

4 other bachelor students work on the same research topic to give an answer to the 4th RQ. Half of them analyse the performance of the conventional ML models Random Forest, SVM and k-NN on two other datasets, while the other half uses DNNs (CNN and LSTM) to assess the classification accuracy on all three available datasets: SedentaryActivity, DesktopActivity and ReadingActivity. The difference between the dataset used in this research and the other datasets is that they have only 5 desktop-related activities and just 8 subjects, resulting in less data for classification especially for the conventional ML models. Nevertheless, there are some similarities between the obtained classification scores. The Random Forest classifier, identified as the most reliable of all conventional ML models applied in this research, is used for cross-dataset comparison. Firstly, for person-dependent training, the best score in this study is 0.94, corresponding to 0.96 (ReadingActivity), 0.92 (DesktopActivity), 0.98 (LSTM) and 0.99 (CNN). Secondly, for person-independent training, this research finds that Random Forest has accuracy of 0.65, compared to 0.67 (ReadingActivity), 0.58 (DesktopActivity), 0.67 (LSTM) and 0.69 (CNN). Random Forest performs best on SedentaryActivity and ReadingActivity across all activities, whereas the SVM performs best on DesktopActivity and ReadingActivity by subjects. Overall, for person-independent training, the three conventional ML models and the two DNNs have accuracy between 60 and 70% which are lower than the person-dependent scores due to the heterogeneity in human visual behaviour. When it comes to the memory usage of the ML

models and the DNNs, the latter takes much more memory for processing since DNNs have many layers of computation, store the input data and their parameters and activations when training and fitting the data in comparison to conventional ML models [19, 20].

4 Responsible Research

This chapter discusses the ethical aspects of this study (4.1) and the reproducibility of its results (4.2).

4.1 Scientific Integrity

The dataset used in this research is collected by N. Srivastava, J. Newn and E. Velloso [2]. It contains raw gaze coordinates which are hard to interpret, but once processed, the data analysis becomes much more comprehensible and representative of the subject's visual behaviour. The combination of fixations and saccades together with the screen recording of the activities performed by the subjects give insights into where the participants are looking at and what they are thinking about. This is regarded as "privacy-invading" and might even lead to re-identification of the participant as "eye tracking data may implicitly contain information about a user's biometric identity, gender, age, ethnicity..." according to [21]. However, the dataset does not contain the screen recording, neither does it contain any sensitive information about the participants. After plotting the extracted fixations and saccades, one could deduce where the person focuses his/her attention (e.g. where the number of fixation is large) but there is no danger of tracing that back to the person performing the activity.

4.2 Reproducibility

The work done in this research can be reproduced by following the details explained in sections 2 and 3 and reading the referenced literature carefully. The techniques used for preprocessing the data are explained in section 2.1 together with the details related to extracting the low-level gaze features 2.2. The specifics related to the classifiers are explained in chapter 3. When the cross validation was performed together with the hyperparameter tuning of the person-dependent evaluation, the k-NN classifier always produced optimal results when k=1. However, due to the limitation of the k-NN model mentioned in section 3.3.1, it adapts to the training data specifically which results in a decline of its level of generalizability, which usually means overfitting [22]. However, the cross validation of the person-independent evaluation produced a higher number for k (k=13) which is inline with the findings of Srivastava, Newn and Velloso [2] (k=10). Therefore, k=13 has been used for the person-dependent analysis as well. The best hyperparameters are summarized in Table 2. Details about how to find the optimal features to improve the classification accuracy are given in section 3.3.2.

5 Conclusions and Future Work

This paper uses data obtained from an eye tracker to build three conventional ML classifiers, Random Forest, SVM and k-NN, and assesses their capability of predicting sedentary activities correctly. The primary goal of this research was to find the features corresponding to good classification accuracy. The majority of them are based on the saccadic eye movements and the most important of the fixation-based features turned out to be fixation dispersion area, fixation radius (the newly proposed feature) and fixation slope. The saccade-based features encode the most relevant information that can be captured from the preprocessed data which increases drastically the classification accuracy of the SVM and k-NN classifiers: the scores of 0.26 and

0.32 jump to 0.86 and 0.77 for SVM and k-NN, respectively. The results of the Random Forest classifier remain equally high before and after the feature selection has been applied which makes its performance the most reliable: 0.94. Though, there is one limitation in the separation of data for the person-dependent evaluation (80% of the data of each person is used for training and the remaining 20% - as test set). The choice of activities which end up in the training set is at random which ensures that the classifiers will be trained on all activities but does not mean that 80% of each activity appears in the training which create data imbalance. This can be applied as a future step of this research. The reason why the classifiers perform so well on the person-dependent evaluation is because they are tested on subjects they have been trained on.

Therefore, the bigger challenge is when the system is tested on subjects that have not been used for learning. So, a leave-one-subject-out evaluation has been performed as well. This time, the system is trained on 23 subjects and tested on the remaining one which resembles a real-world scenario more closely. The resulting recognition accuracies are lower than the person-dependent evaluation: 0.65, 0.53 and 0.48 for the Random Forest, SVM and k-NN, respectively. This is because the classifiers learn many different visual behaviours, due to the heterogeneity of human visual behaviour, and are less accurate in their predictions on unseen data from other subjects. The most intriguing score in the accuracy per subject is subject 10, whose scores realize the best results in the person-independent performance analysis. But this score is an exception due to a inconsistencies in the data. For some subjects there are gaps in the benchmarking of when each gaze point has been recorded by the eye tracker. Some signals have large gaps between two consecutive data points in the timestamp and subject 10 is an example of that, which decreases the number of sample observations generated for this participant used in the classification. These cases are just a couple so the overall performance in the two kinds of evaluation is not affected.

To gain a better understanding of the heterogeneity in human visual behaviour, as a next step of this research one could explore different subsets of participants for training and test on multiple participants. This could help identify some common patterns in the visual behaviour and lead to increasing the generalizability of the ML models.

This research is based on Srivastavan, Newn and Velloso's work [2] but it proposes a dynamic way of calculating the thresholds in the fixation filter used for the feature extraction step (1st RQ), finds the most relevant features which lead to better classification accuracy using the mRMR feature selection method (2nd RQ), examines the impact of the heterogeneity in human visual behaviour on the classification accuracy (3rd RQ) and compares the obtained recognition scores across different datasets and using DNNs (4th RQ).

References

[1] K. Kunze, M. Iwamura, K. Kise, S. Uchida, and S. Omachi, “Activity recognition for the mind: Toward a cognitive "quantified self",” 2013.

[2] N. Srivastava, J. Newn, and E. Velloso, “Combining low and mid-level gaze features for desktop activity recognition,” 2018.

[3] K. Kunze, H. Kawaichi, K. Yoshimura, and K. Kise, “The wordometer â estimating the number of words read using document image retrieval and mobile eye tracking,” 2013.

[4] A. Bulling, D. Roggen, and G. Troester, “What’s in the eyes for context-awareness?” 2011.

[5] A. Bulling, J. A. Ward, H. Gellersen, and G. Troester, “Eye movement analysis for activity recognition using electrooculography,” 2010.

[6] K. Kunze, Y. Utsumi, Y. Shiga, and K. Kise, “I know what you are reading â recognition of document types using mobile eye tracking,” 2013.

[7] G. Lan, B. Heit, T. Scargill, and M. Gorlatova, “Gazegraph: Graph-based few-shot cognitive context sensing from human visual behavior,” 2020.

[8] P. Olsson, “Real-time and offline filters for eye tracking,” 2007.

[9] TobiiPro, “Tobii pro x2 eye tracker,” 2022.

[10] P. Blignaut, “Fixation identification: The optimum threshold for a dispersion algorithm,” 2009.

[11] T. R. N. and R. Gupta, “Feature selection techniques and its importance in machine learning: A survey,” 2020.

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

[13] F. L. Hanchuan Peng and C. Ding, “Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy,” 2005.

[14] H. Peng, F. Long, and C. Ding, “Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2005. [Online]. Available: https://doi-org.tudelft.idm.oclc.org/10.1109/TPAMI.2005.159

[15] S. S. Ibrahim, Y. Wang, M. A. Athuman, and X. Fu, “Remote eye gaze tracking research: A comparative evaluation on past and recent progress,” *Electronics*, vol. 10, no. 23, p. 3165, 2021. [Online]. Available: https://www-proquest-com.tudelft.idm.oclc.org/scholarly-journals/remote-eye-gaze-tracking-research-comparative/docview/2612764513/se-2

[16] A. T. Duchowski, “Al-Qaeda, oil dependence, and U.S. foreign policy,” in *Eye Tracking Methodology.* Springer Cham, 2003, pp. 254–256.

[17] W. Koehrsen, “Hyperparameter tuning the random forest in python,” 2018.

[18] P. Kiefer, I. Giannopoulos, and M. Raubal, “Using eye movements to recognize activities on cartographic maps,” 2013.

[19] T. Sarkar, "How much memory is your ml code consuming?" 2021. [Online]. Available: https://towardsdatascience.com/how-much-memory-is-your-ml-code-consuming-98df64074c8f

[20] J. Hanlon, "Why is so much memory needed for deep neural networks?" 2017. [Online]. Available: https://www.graphcore.ai/posts/why-is-so-much-memory-needed-for-deep-neural-networks

[21] J. L. Kröger, O. H.-M. Lutz, and F. Müller, *What Does Your Gaze Reveal About You? On the Privacy Implications of Eye Tracking.* Cham: Springer International Publishing, 2020, pp. 226–241. [Online]. Available: https://doi.org/10.1007/978-3-030-42504-3_15

[22] S. Yildirim, "How important is the k in knn algorithm," 2020. [Online]. Available: https://towardsdatascience.com/how-important-is-the-k-in-knn-algorithm-3b6fce726110

A Normalization formula

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Figure 12: The formula used to normalize each of the coordinate arrays.

B Fixation filter

```python

def estimate_fixations(x, y, peak_indices):
    """
    Estimate the positions of fixations using the median and join fixations
    that are spatially closer than the given threshold radius.

    :param x: List with x coordinates for the current activity.
    :param y: List with y coordinates for the current activity.
    :param peak_indices: The indices of the identified saccade peaks.
    :return: The estimated fixation positions.
    """

    fixation_positions = {
        "x": [],
        "y": []
    }
    distances = []

    for i in range(1, len(peak_indices)):
        fixation_positions["x"].append(np.median(x[peak_indices[i - 1]:peak_indices[i]]))
        fixation_positions["y"].append(np.median(y[peak_indices[i - 1]:peak_indices[i]]))

    for i in range(1, len(fixation_positions["x"])):
        # Calculate the euclidean distance between the current and the previous fixation estimate.
        distance = math.dist([fixation_positions["x"][i], fixation_positions["y"][i]],
                             [fixation_positions["x"][i - 1], fixation_positions["y"][i - 1]])
        distances.append(distance)

    # Sort the distances.
    distances.sort()

    # Compute the index of 10% of the data to merge only 10% of the fixations.
    index = int(0.1 * len(distances))

    # Calculate the threshold radius.
    threshold_radius = distances[index]

    return threshold_radius
```

Listing 2: Estimating fixation positions.

C Precision and recall

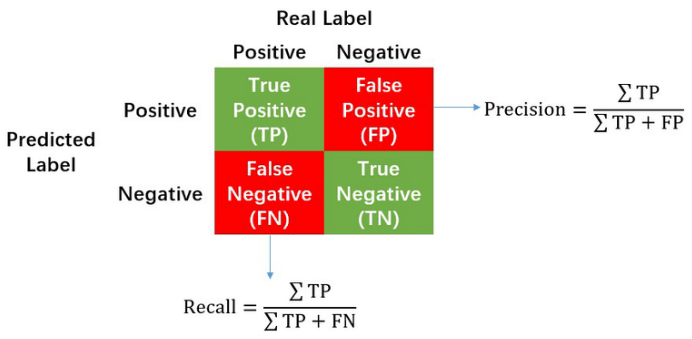
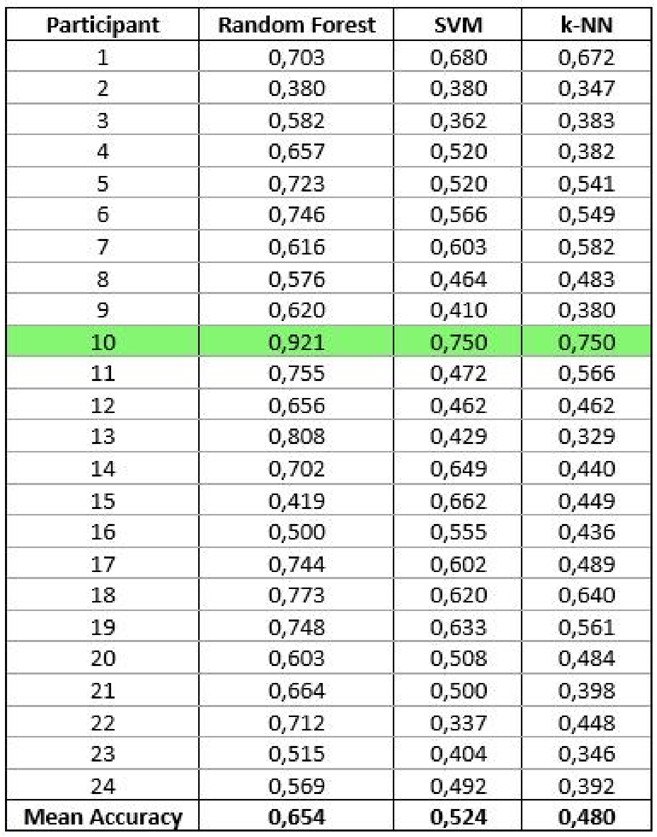


Figure 13: The formulas for precision and recall to evaluate the classifier's performance per activity.

D Classification accuracy for person-independent analysis



Participant	Random Forest	SVM	k-NN
1	0,703	0,680	0,672
2	0,380	0,380	0,347
3	0,582	0,362	0,383
4	0,657	0,520	0,382
5	0,723	0,520	0,541
6	0,746	0,566	0,549
7	0,616	0,603	0,582
8	0,576	0,464	0,483
9	0,620	0,410	0,380
10	0,921	0,750	0,750
11	0,755	0,472	0,566
12	0,656	0,462	0,462
13	0,808	0,429	0,329
14	0,702	0,649	0,440
15	0,419	0,662	0,449
16	0,500	0,555	0,436
17	0,744	0,602	0,489
18	0,773	0,620	0,640
19	0,748	0,633	0,561
20	0,603	0,508	0,484
21	0,664	0,500	0,398
22	0,712	0,337	0,448
23	0,515	0,404	0,346
24	0,569	0,492	0,392
Mean Accuracy	**0,654**	**0,524**	**0,480**

Table 6: The full statistics on the classification accuracy obtained per subject per classifier.