How to maximize the capabilities of in-mouth sensors for human activity recognition?

Maosheng Jiang

Supervisor(s): Koen Langendoen, Hayley Hung, Vivian Dsouza, Stephanie Tan

EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 23, 2024

Name of the student: Maosheng Jiang
Final project course: CSE3000 Research Project
Thesis committee: Koen Langendoen, Hayley Hung, Vivian Dsouza, Stephanie Tan, Qun Song

Abstract

Human activity recognition plays an interesting and important role nowadays as there are a variety of use cases. It is utilized in health monitoring, in the development of human-computer interaction system and in security monitoring. However current methods involve usage of privacy sensitive data and impractical sensors for everyday usage. To tackle this problem, we aim to answer the research question "How to maximize the capabilities of in-mouth sensors for human activity recognition?". The main contributions of this paper are the classification of different gestures using an in-mouth device, implementation of a classifier directly onto a microcontroller and the evaluation whether the models can generalize to multiple people. To investigate this, we experimented with popular classical machine learning classifiers: Decision Tree, K-Nearest Neighbors, Support Vector Machine, Logistic Regression and Random Forest classifiers. The results shows that the F1-score of all classification problems are above 80% using the various classifiers along with different parameters.

Keywords— Human Activity Recognition, In-Mouth Sensor, IMU, Machine Learning

1 Introduction

The field of Human Activity Recognition (HAR) deals with the automatic identification of human activity using sensor data [1]. This area of research has been experiencing rapid development and increasing interest, due to its beneficial impact on human life. For instance, HAR has been applied in health monitoring systems [2, 3], enabling the tracking of patient's health and early detection of medical conditions. Furthermore, HAR is also interesting in the development of human-computer interface systems [4], enabling interaction with computer systems using only gestures. Lastly, it is utilized in security monitoring [5], ensuring security and safety in various environments.

However, existing research in HAR frequently depend on wearable sensors mounted on various parts of the human body [6, 7]. While effective, these wearables can be cumbersome and uncomfortable to wear, making them impractical in real-life scenarios where ease and comfort are essential. Furthermore, other research relies on the processing of camera data [8], which raises substantial privacy concerns.

To address these challenges and difficulties, our research will investigate the efficiency of utilizing an in-mouth sensor for HAR. With this research, we aim to explore a less invasive and more practical alternative for HAR.

The focus of this research paper is to investigate the efficiency of using an in-mouth sensor to perform human activity recognition. Consequently, the research question of this research paper is:

How to maximize the capabilities of in-mouth sensors for human activity recognition?

The efforts of this research paper will contribute to the academic research field in the area of HAR by investigating and exploring the potential applications of using in-mouth sensors.

This paper makes three significant contributions. First, it introduces a method for detecting five human behaviors using an in-mouth sensor: identifying the head position during sleep, detecting speech, predicting whether the sensor is in use while walking or lying flat, detecting whether the sensor is inside or outside the mouth, and detecting whether the mouth is open or closed. Furthermore, the second contribution is the investigation of the feasibility to implement classifiers directly on the hardware using low-resource microcontrollers. Finally, it evaluates whether the models can generalize across multiple individuals or if they are specific to each person.

This paper is structured as follows: Section 2 talks about the related work. Section 3 describes the experimental setup. Section 4 outlines the methodology. The results are then presented and discussed in Section 5. This paper will then be concluded in Section 6 along with a discussion about future work in section 7. Section 8 reflects on responsible research aspects.

2 Related work

Various researchers have conducted studies on HAR using sensor data. There is a limited amount of studies performed on the usage of in-mouth sensors for HAR. Furthermore, there are currently no studies which implements the classification algorithm directly onto the embedded microcontroller (MCU).

2.1 Human Activity Recognition

In the field of HAR, machine learning and deep learning models have been deployed to predict human behavior [9]. The goal is to automate the classification of human behavior, such as walking and jumping. Machine learning models use handmade features and relies on mathematical techniques to construct meaningful features for classification. On the other hand, deep learning techniques can automatically extract meaningful features and patterns from the data without the need to construct features manually. Consequently, deep learning methods generally require much more training data compared to machine learning models [9].

2.2 In-Mouth Sensors

Cascon et al. proposed a novel in-mouth sensor designed to function as a human-computer interface [4]. This research demonstrates the potential of an in-mouth device that senses various variables inside the mouth. To differentiate between different gestures that can be performed inside the mouth using the tongue, it was necessary to train machine learning models to classify the gestures. The classification had a recall score of 97.1% using a Random Forest classifier. This shows promising results of the accuracy of classification using data collected from in-mouth sensors. However, the classification was performed externally and not on the embedded hardware of the in-mouth sensor itself.

2.3 HAR On Embedded Systems

In existing literature about human activity recognition, the training and classification process is executed in the cloud or an external computer [10]. This is due to the fact that the hardware inside the sensor does not have enough computing power to train a machine learning model or make a prediction using input data obtained from the sensors.

The work of Stolovas et al. shows promising results of HAR on an embedded system [6]. The MCU utilized in this paper has 16KB flash memory and 512B RAM. The embedded software was designed to predict three activities: running, standing still and walking. To achieve this, Stolovas et al. applied dimensionality reduction along with a Support Vector Machine classifier. The parameters were trained on an external computer and are later hard coded in the flash memory of the MCU. With an embedded software utilizing only 6 KB flash memory and 240 B RAM, Stolovas et al. managed to achieve an average precision of 97.92%.

In the work of Elsts et al., researchers were able to fit a Convolutional Neural Network (CNN) inside an MCU using the TensorFlow lite library [11]. However, this study and the study of Stolovas et al. both used an experimental setup that is infeasible for practical standalone in-mouth classifications, as the devices requires external power supply and would not fit in the mouth.

3 Experimental setup

This research paper aims investigate the efficiency of using in-mouth sensors for HAR. We have performed various experiments in order to answer this research question. This section will explain the hardware of the in-mouth sensor, the dataset and the conducted experiments.

3.1 Hardware Specifications

The in-mouth sensor that has been used for this research paper consists of various electronics on a flexible Printed Circuit Board (PCB). This PCB is then securely integrated into custom-tailored braces which fits onto the recipient's lower jaw. A top-view of the development board of the sensor is shown in Figure 1.

The in-mouth sensor is equipped with different sensors onboard. For sensing the accelerations along the 3-axis, the device contains a low-power Inertial Measurement Unit (IMU). The IMU additionally includes temperature sensing capabilities as it contains a temperature sensor built-in. This reduces the footprint of the device significantly as there is no need for additional hardware for measuring the temperature. Lastly, it contains a photo-diode that is capable of sensing the light intensity inside the mouth.

The in-mouth sensor contains capacitors which supplies the system with power, removing the need of an external power supply. There is also a NFC interface on-board, utilized for data transfer and charging.

3.2 Dataset

The dataset that has been utilized for this research was collected by the TU Delft Socially Perceptive Computing Lab (SPCL). To collect the data, a script was utilized to instruct

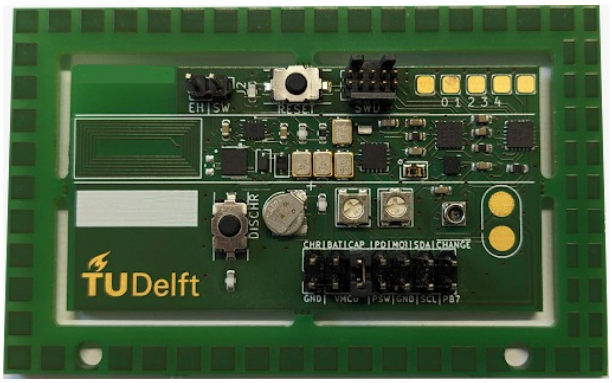

Figure 1: Top view of the development board of the in-mouth sensor.

participants to perform certain actions. The order of actions was randomized to avoid any bias. Three participants used the sensors according to the protocols to collect the labelled data.

The dataset contains 4950 labeled samples from the in-mouth sensor. Each sample contains the following measurements: temperature, light intensity, x-acceleration, y-acceleration, z-acceleration, and the voltage of the in-mouth sensor. The dataset was shared using the Comma-Separated Values (CSV) file format. A visualization of a CSV file is depicted in Figure 3. Figure 4 shows the distribution of the dataset among the three different persons. Person 0 is a 'dummy' person used for activities that do not require the sensor to be in the mouth. Person 1, 2, and 3 are real persons.

An important detail of the dataset is the sampling frequency that has been used to collect the data. For this particular dataset, a sampling frequency of 1 Hz has been used. For HAR, this is a rather low sampling frequency as discussed in [12]. In this work, it is shown that popular public benchmark datasets for HAR have a sampling frequency ranging from 20 Hz to 250 Hz. This has a significant impact on behavior that is dependent on time-series data, as there will be loss of information when capturing human behavior using a low sampling frequency.

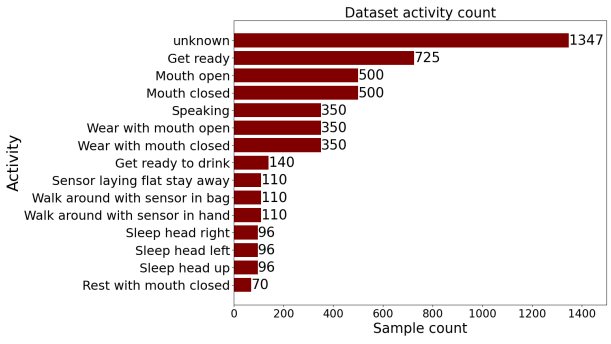

Figure 2: The amount of samples for each activity in the dataset.

3.3 Experiments

For the experiments, we first explored what kind of different human behavior we could classify based on the provided

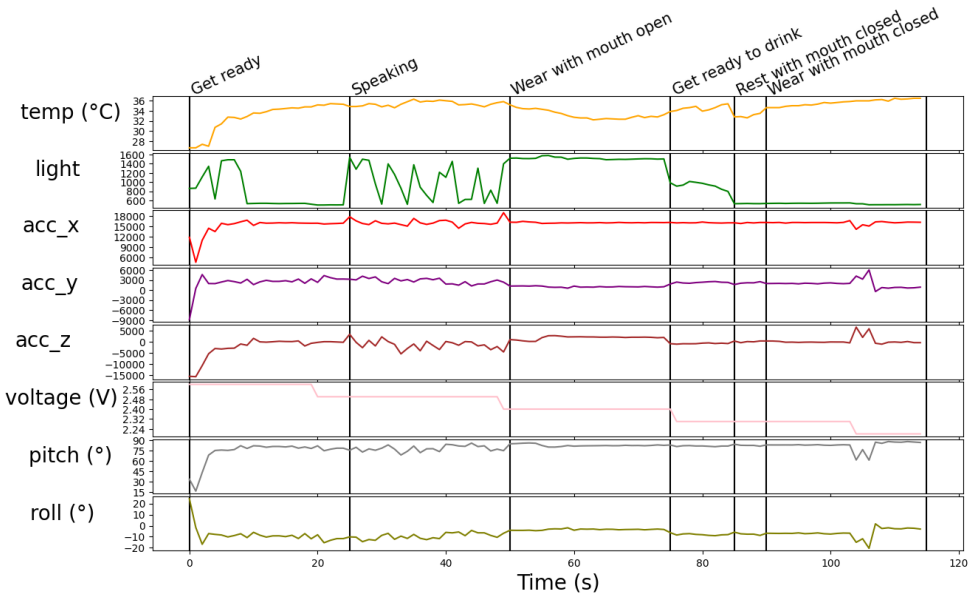

Figure 3: An example of data collected during various activities. The pitch and roll values are calculated based on the x-, y-, and z-accelerations and were not present in the original dataset.

dataset. The final behaviors to be classified were as follows:

- Predicting the head position during sleep.
- Predicting whether the person using the in-mouth sensor is speaking or not.
- predicting whether the sensor is in use while walking or lying flat.
- Predicting whether the in-mouth sensor is inside or outside the mouth.
- Predicting whether the mouth is open or closed.

Furthermore, for each of these classification problems, we preprocess the dataset to create training data. This data is subsequently used to train machine learning models to predict the behavior. Then, we compare and evaluate the performance of the different machine learning models.

Finally, we experimented with the implementation directly on a MCU using the popular and affordable STM32 Blue Pill development board. This board features the STM32F103C6T6 MCU, which includes 32KB of flash memory and 10KB of RAM. To visualize the predicted classes, we used 3 Light Emitting Diodes (LED), one for each class. The setup, as depicted in Figure 5, was constructed on a breadboard and includes an MPU6050 IMU with an integrated temperature sensor, as well as a Light Dependent Resistor (LDR) to measure light intensity.

This experimental setup differs from the hardware used in the actual in-mouth sensor as shown in Figure 1. This setup was chosen to facilitate rapid prototyping, allowing for easy adjustments and testing. The most notable differences are that the actual MCU on the in-mouth sensor has 16KB of flash memory and 2KB of RAM.

4 Methodology

This section deals with the technical side of the conducted experiments, such as the details of the preprocessing step, the machine learning models and the parameters that were used. The experiments were carried out in the Python programming language, chosen for its ease of use and extensive library ecosystem. Additionally, the `NumPy` library was used for data processing and manipulation [13].

4.1 Data Preprocessing

In order to create training data suitable for analysis, it is necessary to clean and prepare the raw data from the dataset.

The first step is to filter out samples which are not useful for the classification problems described earlier. The raw dataset contains two labels which will not be used for training: the label ”Unknown” and ”Get Ready”. The ”Unknown” label refers to behavior that is not part of a human activity, such as the transition period between performing two different activities. Next, the ”Get Ready” label is used for the initial startup phase of the in-mouth sensor, where the device needs to be prepared for collection of data and to warm up the temperature sensor to body temperature.

Secondly, we need to balance the dataset. The dataset consists of samples which are labeled with various activities. However, each activity has a different number of samples in the dataset, as can be seen in figure 2. The usage of imbalanced training data can lead to a biased model and reduced performance. To tackle this, we applied downsampling of the majority class. This means that if we have a binary classification problem and the training data consists of 1200 samples of class 1 and 500 samples of class 2, then the final training data will contain 1000 samples, 500

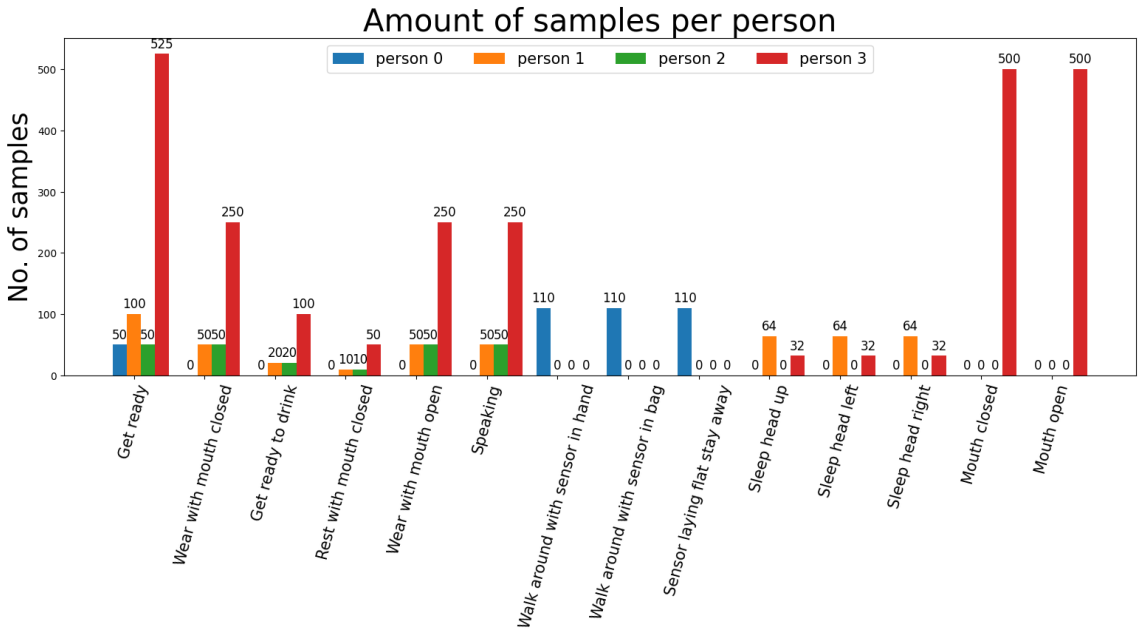

Figure 4: Dataset distribution across persons.

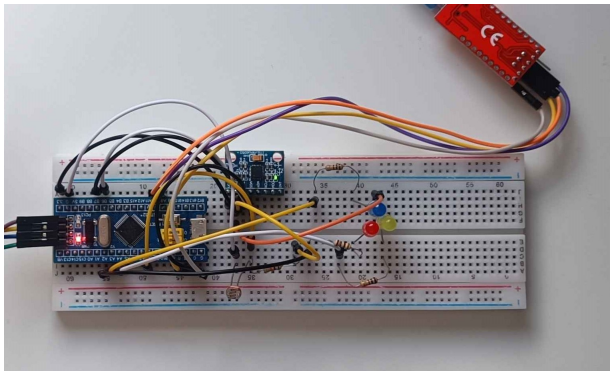

Figure 5: Experimental breadboard setup.

of class 1 and 500 of class 2. The down-sampling is performed with the `RandomUnderSampler` class provided in the `imbalanced-learn` Python library [14].

4.2 Feature Extraction

The third step involves extracting features from the clean raw data in order to create feature vectors for training purposes. As the raw dataset contains five variables (temperature, light intensity, x-, y-, and z-accelerations), we calculated two additional variables for each sample, the pitch and roll angles, using Equations 1 and 2 respectively.

$$pitch = arctan\left(\frac{A_x}{\sqrt{(A_y)^2 + (A_z)^2}}\right) \quad (1)$$

$$roll = arctan\left(\frac{A_y}{\sqrt{(A_x)^2 + (A_z)^2}}\right) \quad (2)$$

In Equations 1 and 2, the A_x, A_y, and A_z variables represent the accelerations on the x-, y-, and z-axis respectively.

The fourth and final step is to apply a sliding window approach for the time-series data in order to predict dynamic behavior that is dependent on time, e.g. speaking and walking. Using a sliding window approach of 2s or 3s windows and around 50% overlap has shown to be effective for the purpose of human activity recognition in past research [15, 16].

To extract features from a window, we considered using time- and frequency-domain analysis. These were motivated by past research in the topic of feature extraction for human activity recognition using accelerometer data [16, 17].

Frequency-domain analysis of a window can be performed using for example Fast Fourier Transforms (FFT) or Wavelet Transforms (WT). These kind of transforms unfortunately do not work well with out dataset that is sampled at 1 Hz. For instance, Preece et al. suggests a window size of at least 128 samples for wavelet transforms in order to effectively decompose into wavelet coefficients [16].

Since frequency-domain analysis is infeasible, we instead utilize time-domain analysis for feature extraction of our windows. The features that are extracted are the Mean (μ) and Standard Deviation (σ) from the window.

The sliding window approach was implemented iteratively, looping through each sample of the dataset corresponding to the same activity. This means there is are no window processed where two or more different activities were performed.

4.3 Classifiers

To be able to classify the different behaviors, we utilized several popular classical machine learning algorithms. We used Decision Trees (DT), K-Nearest Neighbors (KNN), Support Vector Machines (SVM), Logistic Regression (LR), and Random Forests (RF), which have also been used in existing work

as well [15, 18].

Although other types of machine learning, such as deep learning, show promising results [19], we did not consider them for this research due to their high computing power and resource demands. One of the objectives of this research is to implement the classification directly on the embedded microcontroller of the in-mouth sensor. The MCU of this specific in-mouth sensor lacks the necessary resources and computing power to support a Neural Network. Therefore, for our experiments, we utilized the five previously mentioned machine learning algorithms.

4.4 Parameters

For the machine learning models presented in the previous section, we utilized the implementations available from `scikit-learn`, a Python library for machine learning [20].

Since the performance of machine learning models depends on various parameters, we experimented with different settings for each model. For the Decision Tree, we tested tree depths of 1, 2, 3, 4, 5, 10, 15, and 20. For the K-Nearest Neighbors, we normalized the data and tested using 1, 2, 3, 4, 5, 10, 15, and 20 neighbors. For the Support Vector Machine, we evaluated the polynomial, linear, and radial basis function (rbf) kernels. For Logistic Regression, we used the default settings provided by `scikit-learn`. Lastly, for Random Forests, we tested tree depths of 1, 2, 3, 4, 5, 10, 15, and 20, combined with 10, 20, 30, 40, 50, and 100 trees.

For the identification of the head position during sleep, we utilized the labels as shown in table 1. The features that were used are x-, y-, and z-acceleration values. There was no sliding window approach applied since one sample is enough for the prediction.

For detecting speech, we utilized the labels as shown in table 2 We used a sliding window approach as speaking is a dynamic behavior. The features from the window were the mean and the standard deviation of the x-, y-, and z-accelerations.

For identifying whether the in-mouth sensor is being used during walking or while laying flat, the labels that were utilized are shown in table 3. The sliding window approach has been applied here as well as walking is a dynamic behavior. Here the features that were extracted from the windows were the mean and standard deviation from the x-, y-, and z-accelerations.

In the case of detecting whether the in-mouth device is inside or outside the mouth, we categorized the labels as shown in table 4. Here, the prediction is made from a single sample. The only feature utilized is the temperature measured by the in-mouth sensor. This is because the temperature inside the mouth is higher than the temperature outside.

Lastly, for the detecting whether the mouth is closed or open, the labels were utilized as shown in table 5. In this case, the prediction is also made from a single sample. The utilized feature is light intensity, as it increases when the mouth is open.

4.5 Evaluation

To evaluate the classifier models, we used K-fold cross validation. The choice of 10 is common in literature [1, 7]. Additionally, 5 is another common choice for K [3, 17]. There is no universal rule for the choice of K, as the choice is highly context dependent. We have chosen to use 10-fold cross validation to provide more training data to our model. Using K=10 over K=5 means that we will have 90% training data and 10% test data in each iteration instead of 80%/20%. This will maximize the usage of the available data, providing a more reliable performance.

class 0	class 1	class 2
Sleep head up	Sleep head left	Sleep head right

Table 1: The dataset labels that were used for training in the case of predicting the head position during sleep.

class 0	class 1
Speaking	Wear with mouth closed
	Wear with mouth open
	Walk around with sensor in hand
	Walk around with sensor in bag
	Sensor laying flat stay away
	Sleep head up
	Sleep head left
	Sleep head right
	Mouth closed
	Mouth open

Table 2: The dataset labels that were used for training in the case of detecting speech.

Another validation method used in human activity recognition is the Leave-One-Subject-Out Cross Validation (LOSOCV) [11]. This method utilizes training data from all subjects except one and using the data from the left-out subject for testing. However, in our study, we have data from only 3 subjects, with varying amounts of data collected from each person. As a result, using LOSOCV will not be reliable in this case.

4.6 Implementation on MCU

Since the IMU measurements are dependent on the sensor's mounting location and our experimental breadboard setup could not fit in the mouth, we opted to simulate sampling by sending the dataset over USB serial. This approach mimics the data sampling that would have been done by the actual in-mouth sensor. Each data sample is transmitted at 1-second intervals to match the 1 Hz sampling frequency, and includes five values: light intensity, temperature, and x-, y-, and z-accelerations.

The embedded code on the MCU, written using the Arduino framework for its ease of use and simplicity, parses the incoming values into a buffer and makes predictions based on the features calculated from the buffer. It calculates the pitch and roll angles on board. In the case of a sliding window, the windows are stored on board, where the mean and standard deviations are also computed.

We experimented with the Decision Tree classifier, as it is the most feasible model to implement on a low-power MCU with 16KB flash and 2KB RAM compared to the other models. Furthermore, the actual available flash size and RAM are further reduced due to the necessary code for booting and communicating with peripherals such as the IMU. The Decision Tree is implemented by converting the trained Decision Tree model from `SciKit-Learn` into nested if-else

class 0	class 1
Sensor laying flat stay away	Walk around with sensor in bag
	Walk around with sensor in hand

Table 3: The dataset labels that were used for training for the identification of whether the sensor is in use while walking or lying flat.

class 0	class 1
Wear with mouth closed	Walk around with sensor in hand
Wear with mouth open	Walk around with sensor in bag
Mouth closed	Sensor laying flat stay away
Mouth open	

Table 4: The dataset labels that were used for training in the case of detecting whether the in-mouth device is inside or outside the mouth.

statements. The nested if-else statements is then used for the prediction of the behavior.

To evaluate the embedded software, we compiled the code both with and without the classification-related code. The components responsible for classification include functions for feature calculation, buffers for the sliding window approach, and nested if-else statements representing our Decision Tree. The difference in size between the two compiled binaries indicates the memory required for classification.

5 Results and discussion

In this section, the results of the machine learning models are shown using figures. The metric used to compare the different parameters of the classifier is the F1-score. This score combines precision and recall into one number, making it easier to see how well the model balances correctly identifying positive results and avoiding false positives.

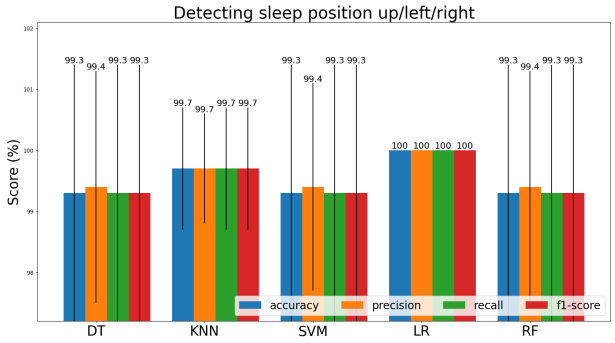

Figure 6: Performance of models trained for classifying the sleeping position of the head. DT depth = 5, KNN n = 1, SVM kernel = linear, RF depth = 20, RF trees = 30.

5.1 Discussion

For the different classification problems introduced in section 3.3, we tried out different parameters for the models, as described in section 4.4. The results of the models for each classification problem are shown in Figures 6, 7, 8, 9 and 10. These models were trained on the entire dataset containing data from multiple persons.

In Figure 11, the effect of the window size and overlap is illustrated for the detection whether the person is speaking or not. It can be seen that a larger window size leads to improved performance. However, in this scenario, a larger window size may be impractical for real-life applications due to the 1 Hz sampling rate. Although a window size of 5 demonstrates better performance, waiting 5 seconds for a prediction is not feasible since the dynamic behavior may last less than 5 seconds.

class 0	class 1
Wear with mouth open	Wear with mouth closed
Mouth open	Mouth closed

Table 5: The dataset labels that were used for training in the case of detecting whether the mouth is closed or open.

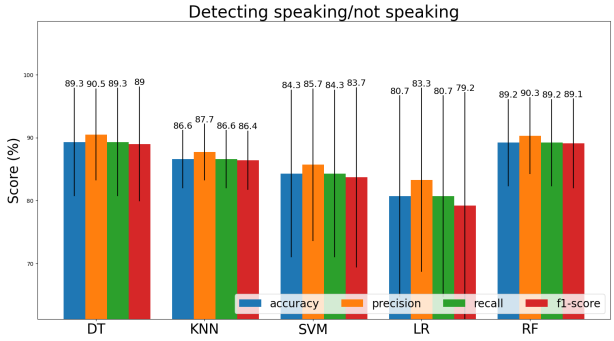

Figure 7: Performance of models trained for detecting speech. DT depth = 2, KNN n = 5, SVM kernel = linear, RF depth = 5, RF trees = 50. Window size = 4 and 1 sample overlap.

In section 4.2 we discussed the usage of the pitch and roll angles as potential features. However our results in Figure 12 show that using the pitch and roll values leads to slightly worse performance compared to x-, y-, and z-accelerations. Therefore, although we have experimented with pitch and roll values, we did not incorporate them them into our final models.

Furthermore, we experimented with training the machine learning models using both the entire dataset, which includes multiple individuals, and separate per-person training. In Figure 13, the results are displayed for detecting the head position during sleep and detecting speech. For head position detection, per-person training with data from person 3 achieves 100% F1-scores for all models. This could be due to variations in IMU placement between individuals, which may impact accuracy when training with multiple individual. For speaking detection, the results are very similar across different training approaches.

Lastly, the results of the implementation directly on the MCU are shown in Figure 14. Detecting speech and predicting whether the in-mouth device is used for walking or is lying flat require additional RAM due to the need to store samples for the sliding windows. Furthermore, all classification problems lie within the constraints of the MCU of the in-mouth sensor in terms of additional flash size and RAM usage.

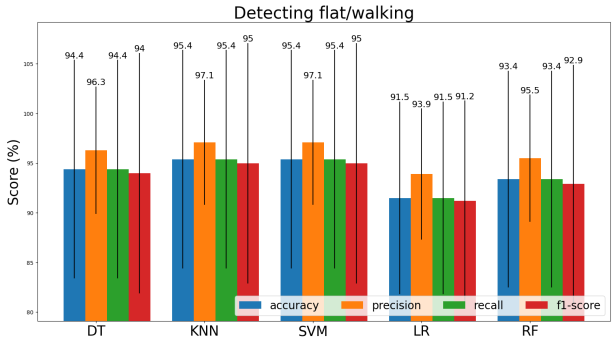

Figure 8: Performance of models trained for predicting whether the in-mouth device is used while waking or whether it is laying flat. DT depth = 1, KNN n = 5, SVM kernel = rbf, RF depth = 1, RF trees = 10. Window size = 4 and 2 sample overlap.

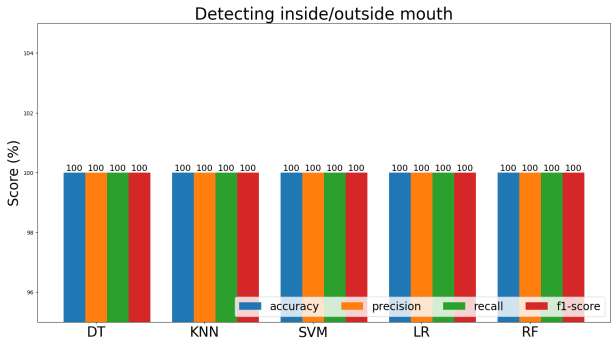

Figure 9: Performance of models trained for predicting whether the in-mouth device is inside or outside the mouth. DT depth = 1, KNN n = 1, SVM kernel = linear, RF depth = 1, RF trees = 10.

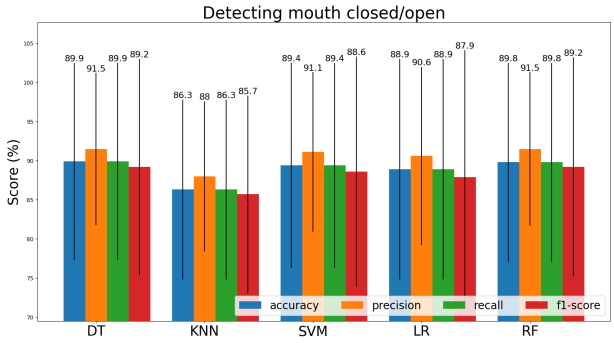

Figure 10: Performance of models trained for predicting whether the mouth is open or closed. DT depth = 1, KNN n = 10, SVM kernel = linear, RF depth = 1, RF trees = 10.

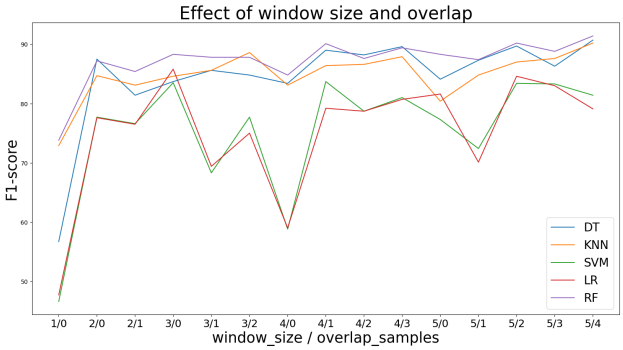

Figure 11: The performance of different classifiers when using different window sizes and overlaps in the sliding window. The classification here is the detection of speech.

6 Conclusions

In this research paper, we have investigated how to maximize the capabilities of utilizing in-mouth sensors for human activity recognition. In particular, we have investigated the efficiency of human activity recognition using data obtained from sensors on the in-mouth sensor. To address this, we established an experimental setup and developed a methodology inspired by an extensive review of previous research.

Based on the research and results, we can conclude that it is possible to efficiently perform human activity recognition using data obtained from sensors on the in-mouth sensor. The findings or this study show promising results, demonstrating that each classification problem achieved an F1-score of 80 or higher using various machine learning models and parameters.

Furthermore, the findings demonstrate that the models show robustness when trained on data collected from in-mouth sensors across multiple individuals. Although there are differences when training the model on a per-person basis, these differences are not significantly greater than when training with data from multiple persons.

The impact of window size is another interesting factor. Window sizes between 2 and 4 provide the best performance while maintaining practical usability in real-life applications. Additionally, the overlap influences the model's performance and should be selected appropriately for each specific problem.

Lastly, implementing a classifier, specifically a Decision Tree, directly on an MCU is feasible. The additional flash memory required fits within the constraints of the in-mouth sensor's MCU, and the RAM usage is also within acceptable limits.

7 Future Work

In this section, we will offer suggestions for future research to build upon this study. The use of in-mouth sensors for human activity recognition is a relatively unexplored field, presenting numerous potential research questions to be addressed.

For future work, the logical next step is to test the implementation of the Decision Tree classifier on the actual in-mouth sensor to evaluate its performance in real-world scenarios. For example, detecting the head position during sleep requires only 80 bytes of additional flash memory and no extra RAM, making it possible to incorporate into future versions of the in-mouth device.

Next, it is interesting to investigate the applications of the capability to predict human behavior. Converting these predictions into signals could transform the in-mouth sensor into an effective human-computer interaction system.

Another area for future improvement is the hardware of the

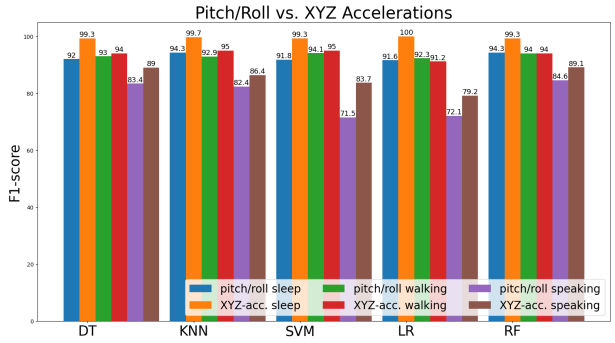

Figure 12: Performance comparison between using pitch/roll values and x-, y-, and z-accelerations for dynamic behavior. Sleep refers to the head position during sleep classification, walking refers to the prediction of whether the in-mouth sensor is used during walking or whether it is laying flat and speaking refers to the detection of speech.

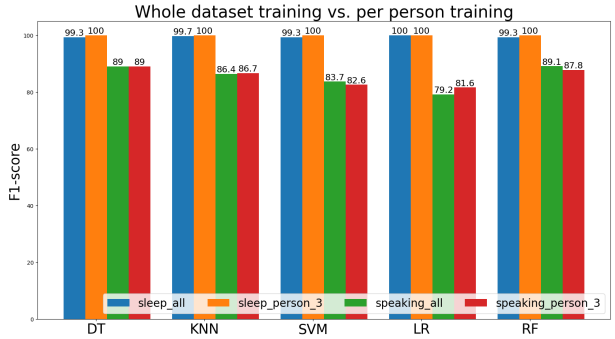

Figure 13: Performance of models trained on the whole dataset compared to training on data only from person 3. Sleep refers to the head position during sleep classification and speaking refers to the detection of speech.

in-mouth sensor utilized in this research. With technological advancements and increasingly faster hardware, it might be possible to increase the current sampling frequency of 1 Hz. A higher sampling frequency would offer the advantage of capturing more complex gestures that can occur within a single second, such as a nod of the head.

Related to the hardware, the current in-mouth sensor could be enhanced by incorporating more sensors. For example, adding a barometer to measure air pressure inside the mouth or a gyroscope to measure yaw. This could lead to a wider range of possible gestures to be recognized.

8 Responsible Research

In this section, we discuss the responsible research aspects of our research. The following subsections will talk about the ethical considerations of our dataset and about the reproducibility of our methods.

8.1 Data Usage

As mentioned before, the dataset that has been used for this research consists of labeled data gathered by 3 subjects. Each instance contains temperature, light intensity, voltage, x-, y-, and z-acceleration readings. This information does not contain privacy sensitive data.

Classification	flash	RAM
Head up / left / right	80	0
Speaking / not speaking	1040	60
walking / flat	408	20
inside / outside	56	0
open / closed	360	0

Figure 14: Additional required flash memory and RAM required (in bytes).

One possible concern about using accelerometer data inside the mouth is the possibility of identifying the person who is wearing the in-mouth sensor. The accelerometer data could be analyzed by malicious parties for the identification of the person based on the unique mouth movements or speech patterns. This could happen if the attacker has access to the IMU data linked to a specific person.

Building upon this, it might be possible to reconstruct the speech from accelerometer data using advanced signal processing techniques. This leads to unintended disclosure of private conversations by malicious parties. This could happen when attackers intercepts or gain access to the accelerometer data.

8.2 Reproducibility

To encourage the reproducibility of this research, we have made all the source code available at our TU Delft GitLab repository[1]. Next, the Experimental Setup and the Methodology sections were written as detailed as possible for reproducibility purposes, highlighting all the different experiments and technical settings. With this detailed explanation of the experiments, we aim to enable others to replicate our findings.

Acknowledgements

I would like to express my gratitude to my direct supervisor, V. Dsouza, for his support throughout the research project. I would also like thank my responsible professor Prof. Dr. K.G. Langendoen for his presence and support in the weekly meetings. Not to forget, supervisors H. Hung and S. Tan provided valuable insights that greatly benefited my research for which I am grateful. Lastly, I would like to thank my group members T. Star, K. Nam, and D. Zhang for their teamwork and dedication throughout the project.

References

[1] Swathi Jamjala Narayanan, Boominathan Perumal, Sangeetha Saman, Aakar Mutha, and Rajen B Bhatt. Human activity recognition on accelerometer data using machine learning algorithms. In *2022 IEEE World Conference on Applied Intelligence and Computing (AIC)*, pages 48–53, 2022.

[1] https://gitlab.ewi.tudelft.nl/cse3000/2023-2024-q4/Hung_Pawelczak_Langendoen_Dsouza/maoshengjiang-Rethinking-ubiquitous-smart-sensing-of-social-beh

[2] Sakorn Mekruksavanich and Anuchit Jitpattanakul. Fallnext: A deep residual model based on multi-branch aggregation for sensor-based fall detection. *ECTI Transactions on Computer and Information Technology (ECTI-CIT)*, 16:352–364, 09 2022.

[3] O. Dehzangi, M. Mohammadi, and Y. Li. Smart brace for monitoring patients with scoliosis using a multimodal sensor board solution. In *2016 IEEE Healthcare Innovation Point-Of-Care Technologies Conference (HI-POCT)*, pages 66–69, 2016.

[4] Pablo Gallego Cascón, Denys J.C. Matthies, Sachith Muthukumarana, and Suranga Nanayakkara. Chewit. an intraoral interface for discreet interactions. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–13, New York, NY, USA, 2019. Association for Computing Machinery.

[5] Anuchit Jitpattanakul Sakorn Mekruksavanich. Automatic recognition of construction worker activities using deep learning approaches and wearable inertial sensors. *Intelligent Automation & Soft Computing*, 36(2):2111–2128, 2023.

[6] Ilana Stolovas, Santiago Sux00E1;rez, Diego Pereyra, Francisco De Izaguirre, and Varinia Cabrera. Human activity recognition using machine learning techniques in a low-resource embedded system. In *2021 IEEE URUCON*, pages 263–267, 2021.

[7] Shuangquan Wang, Gang Zhou, Yongsen Ma, Lisha Hu, Zhenyu Chen, Yiqiang Chen, Hongyang Zhao, and Woosub Jung. Eating detection and chews counting through sensing mastication muscle contraction. *Smart Health*, 9-10:179–191, 2018. CHASE 2018 Special Issue.

[8] Chirag Raman, Jose Vargas Quiros, Stephanie Tan, Ashraful Islam, Ekin Gedik, and Hayley Hung. Conflab: A data collection concept, dataset, and benchmark for machine analysis of free-standing social interactions in the wild, 2022.

[9] Saad AlKharji, Aysha Alteneiji, and Kin Poon. Imu-based human activity recognition using machine learning and deep learning models. In *2023 6th International Conference on Signal Processing and Information Security (ICSPIS)*, pages 62–66, 2023.

[10] Tahera Hossain, Md Shafiqul Islam, Md Atiqur Rahman Ahad, and Sozo Inoue. Human activity recognition using earable device. page 81 – 84, 2019. Cited by: 37.

[11] Atis Elsts and Ryan McConville. Are microcontrollers ready for deep learning-based human activity recognition? *Electronics*, 10(21), 2021.

[12] Ramanujam Elangovan, Thinagaran Perumal, and s Padmavathi. Human activity recognition with smartphone and wearable sensors using deep learning techniques: A review. *IEEE Sensors Journal*, PP:1–1, 03 2021.

[13] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[14] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017.

[15] Ramona Luca, Silviu-Ioan Bejinariu, Hariton Costin, Florin Rotaru, and Gladiola Petroiu. Human activity recognition using inertial data. In *2021 12th International Symposium on Advanced Topics in Electrical Engineering (ATEE)*, pages 1–5, 2021.

[16] Stephen J. Preece, John Yannis Goulermas, Laurence P. J. Kenney, and David Howard. A comparison of feature extraction methods for the classification of dynamic activities from accelerometer data. *IEEE Transactions on Biomedical Engineering*, 56(3):871–879, 2009.

[17] Mohamed Bennasar, Blaine A. Price, Daniel Gooch, Arosha K. Bandara, and Bashar Nuseibeh. Significant features for human activity recognition using tri-axial accelerometers. *Sensors*, 22(19), 2022.

[18] Agus Eko Minarno, Wahyu Andhyka Kusuma, and Rizalwan Ardi Ramandita. Classification of activity on the human activity recognition dataset using logistic regression. *AIP Conference Proceedings*, 2453(1):030003, 07 2022.

[19] Sakorn Mekruksavanich and Anuchit Jitpattanakul. Deep convolutional neural network with rnns for complex activity recognition using wrist-worn wearable sensor data. *Electronics (Switzerland)*, 10(14), 2021. Cited by: 68; All Open Access, Gold Open Access.

[20] 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*, 12:2825–2830, 2011.