Eye-Rubbing Detection Using a Smartwatch

Tom Mery
Ecole Polytechnique Fédérale de Lausanne
Lausanne, Switzerland
tom.mery@epfl.ch

Sina Elahi
Fondation Adolphe de Rothschild
Paris, France
selahi3000@hotmail.com

Alain Saad
Fondation Adolphe de Rothschild
Paris, France
dralainsaad@gmail.com

Damien Gatinel
Fondation Adolphe de Rothschild
Paris, France
gatinel@gmail.com

Alexandre Alahi
Ecole Polytechnique Fédérale de Lausanne
Lausanne, Switzerland
alexandre.alahi@epfl.ch

***Abstract*—In this work, we present a new machine learning method based on the Transformer neural network to detect eye rubbing using a smartwatch. In ophthalmology, the accurate detection and prevention of eye-rubbing could reduce incidence and progression of ectasic disorders such as Keratoconus, and prevent blindness. Our approach leverages the state-of-the-art capabilities of the Transformer network, widely recognized for its success in the field of natural language processing (NLP). We evaluate our method against several baselines using a newly collected dataset and achieve an impressive accuracy of 97% with fine-tuning. Notably, our model operates in real-time on an Apple Watch, enabling prompt detection and response. To facilitate reproducibility, we publicly share our dataset and methods. This research contributes to advancing eye rubbing detection and establishes the groundwork for further studies in hand-face interactions monitoring using smartwatches.**

1. Introduction

Keratoconus is a progressive eye disease that affects the cornea and can cause visual impairment and blindness if left untreated. One of the risk factors for the development and progression of keratoconus is eye rubbing, which can lead to corneal thinning, ectasia and visual loss [1]. Although eye rubbing is a common behavior, its frequency and duration are difficult to measure objectively, which hinders efforts to assess its impact on keratoconus and develop effective interventions. To address this gap, the need for an objective method of Eye-Rubbing detection is evident.

Accurate detection of unconscious daily gestures and movement patterns has many potential applications in habits tracking, hygiene, sports, self-improvement and specifically in healthcare and disease prevention. Key habits to limit transmission of infectious disease like the COVID-19, are face touching avoidance, especially mucosal membranes (eyes, nose, mouth) and regular hand-washing. It has been estimated that face-touching occurs on average 23 times per hour [2]. However although efforts have been made to detect various hand-body interactions [3], [4], [5], face-touching detection represents a challenge due to differentiating hand-to-face proximity (in gestures such as glasses removal, eating / drinking, smoking, hair brushing, or toothbrushing) with actual contact. Moreover, differentiating high-risk mucosal membrane contact with contact with skin, glasses, or clothes is of primordial importance. Researchers have had encouraging results but limited by either technical restraints such as multiple or unpractical sensors, body instrumentation with multiple devices, or artificial constraints such as fixed pre-determined gestures [4], [6], [7]. Some have achieved promising results, up to accurately predicting the specific area of the face that was touched, however without detecting actual contact, resulting in a high proportion of false positives [8]. Although acceptable for preliminary stages of development, in a real-life situation, increased rate of false-positives will undoubtedly lead to users' fatigability, and prevent long-term use of such devices. It is therefore key to further improve the face-touching detection as well as the specificity of the notifications while preserving user-friendliness, ease-of-use, and avoiding hardware encumbrance.

Most studies base their algorithms on the readily available accelerometers, which have shown promises in detecting some gestures such as body-tapping [3] but are limited on their own when it comes to detecting face contact. Other sensors have been investigated to improve results, such as proximity Inertial Measurement Unit (IMU), gyroscopes, or thermosensors, but never achieved expectations for real-life scenario [8], [9], [10], [11]. Impressive results were also obtained from sound, magnetic fields, conduction, or pressure sensors [12], [13], [14], [15]. However these rely on cumbersome devices, or an additional emitting device, worn on the finger for example or even on smart textile or skin-based sensors [16], [17], [18]. Some of the most advanced results were obtained with a wrist-worn device combined with strap-based infrared sensors [19], [20], impedence tomography [21], force-sensitive sensors [22], or photodiodes and LED measuring wrist-contour [23], [24].

These all achieved efficient finger recognition from wrist-based sensors wearable on an every-day watch. In case of face-touching recognition, it is key to be able to correctly identify fingers' subtle gestures and positions. None of these however, can be usable in a daily-life situation.

In an attempt to compromise between state-of-the-art technology, daily-life situation and user-friendliness, we aimed to explore the boundaries of detection using minimally invasive hardware, such as a wristband or smartwatch, to assess the extent of its capabilities. While most current smartwatches offer accelerometer, magnetometer, and gyroscope data, the Apple Watch® (Apple Inc.) stood out as the only readily-available smartwatch that also provided orientation data (roll, pitch, yaw), making it the chosen device for our study.

2. Related Work

In the field of human activity recognition from wearable sensor data, previous research has primarily focused on the classification of general human activities like walking, running, and swimming. End-to-end deep learning based techniques are now widely used as they can simultaneously learn feature representation and classification using supervised training, eliminating the need for manual feature crafting. CNNs and LSTM networks have been extensively used for these tasks. CNNs can extract spatial information, while LSTM networks are well suited at modeling temporal dependencies. Specifically, the combination of CNN with recurrent networks (DeepConvLSTM) [25] has shown notable performances.

In recent years, the application of transformer models in multivariate time series classification tasks, including human activity recognition, has been explored. A transformer network is a type of neural network architecture that has gained significant popularity in the field of natural language processing (NLP). It was introduced in the paper "Attention Is All You Need" by Vaswani et al. in 2017 [26] and has since become a cornerstone of many state-of-the-art NLP models. In the context of human activity recognition, the self-attention mechanism employed by transformers allows them to attend to different elements of the input sequence, enabling them to effectively identify and classify human actions. A self-attention based neural network model that foregoes recurrent architectures showed clear improvements with respect to previous benchmark models (DeepConvLSTM) on four different public datasets [27], [28].

Furthermore, unsupervised pre-training techniques have been successfully applied to multivariate time series classification [29]. By leveraging large amounts of unlabeled data, these techniques enable the model to learn meaningful representations and features, which can be fine-tuned for specific classification tasks.

While previous works have made significant contributions to the field of human activity recognition from wearable sensor data, the specific task of classifying hand-face interactions remains relatively unexplored. Our approach does not introduce new methodologies, it offers a novel application of existing techniques to a specific domain. We demonstrate the applicability and effectiveness of transformer models with self-attention and unsupervised pre-training in the challenging task of classifying specific hand-face interactions with smartwatch's sensors. We also propose different ways of collecting data while ensuring the capture of genuine hand-face interactions in real-world scenarios.

3. Methods

Recall that the purpose of our study is to develop an objective tool for detecting eye rubbing in order to, for example, evaluate its relationship with ectatic corneal diseases such as keratoconus. One of the difficulties, if not the major one in this study, is to succeed in differentiating hand-face interactions that are similar to each other. The solution we propose, therefore, is to develop a machine learning model capable of classifying the various possible hand-face interactions from the sensor's data of an AppleWatch. Fig. 1 illustrates the employed pipeline to achieve the desired outcome.

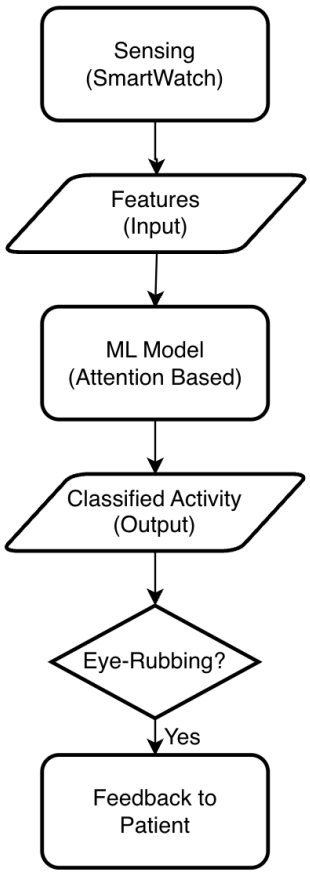

Figure 1. Pipeline

In this section, after presenting the input features and the output classes, we go through the methods used to build the machine learning model.

3.1. Problem Statement

3.1.1. Input. The AppleWatch provides sensor's measures sampled at 50Hz. The signals are composed of the 19 following features provided by the sensors of the AppleWatch:

- Raw Accelerometers Data (see [30]):
  - Acceleration *x,y,z* in G's
- Processed Device-Motion Data (see [31]):
  - Yaw, Roll, Pitch in rad.
  - Rotation Rate *x,y,z* in rad/s.
  - User Acceleration *x,y,z* in G's
  - Quaternion *x,y,z,w*
  - Gravity *x,y,z* in G's

3.1.2. Output. The classes of the classification task are illustrated in Fig. 2.

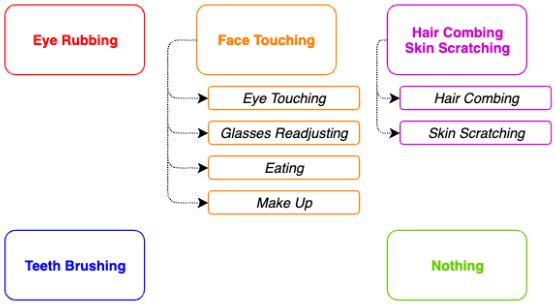

Figure 2. Classes of the classification task

3.1.3. Real-time classification. To enable real-time operation on the Apple Watch, we utilize a sliding window approach illustrated in Fig. 3. This approach involves dividing the continuous stream of sensor data into fixed-size windows. Each window is then processed by the machine learning model, extracting relevant features and performing activity classification to recognize human activities. The window size is set to 3 seconds, with a step size of 0.5 seconds. This configuration allows for classification every 0.5 seconds, utilizing the previous 3 seconds of sensor signals for activity recognition.

3.2. Attention Based Model

The model presented in [27] has been adapted and implemented for the purpose of classifying hand-face interactions. The resulting architecture, depicted in the left part of Fig. 4, incorporates some modifications compared to the original paper. Specifically, the sensor modality attention component has been removed, as our scenario solely relies on data from the AppleWatch sensors. Additionally, we have replaced the simple positional encoding with a learnable positional encoding, which has yielded improved results in our context.

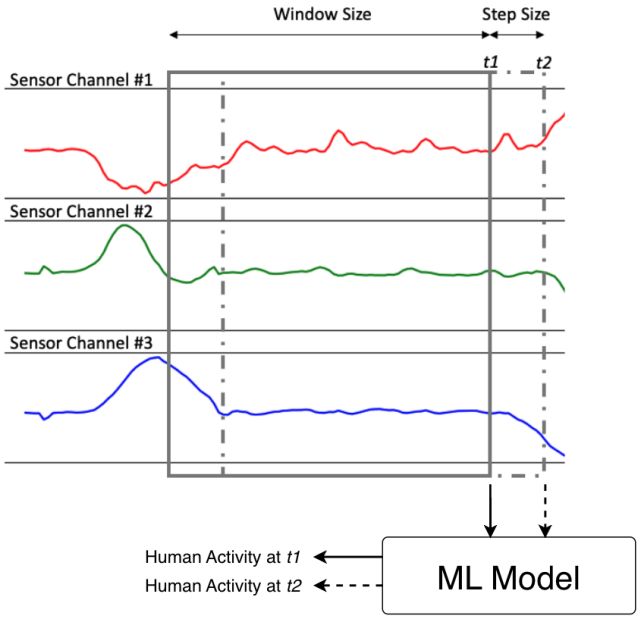

Figure 3. Sliding window approach for real-time classification

3.2.1. Input Encoding. The model receives a time-window of sensor values as input. A linear layer is applied to transform the sensor features. As described in [29], each sample $\mathbf{X} \in \mathbb{R}^{w \times m}$ (multivariate time series of length w with m different variables) constitutes a sequence of w feature vectors $\mathbf{x_t} \in \mathbb{R}^m : \mathbf{X} \in \mathbb{R}^{w \times m} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x_w}]$. The original feature vectors $\mathbf{x_t}$ are linearly projected onto a d-dimensional vector space, where d is the dimension of the transformer model sequence element representations (typically called embedding size):

$$\mathbf{u_t} = \mathbf{W_p}\mathbf{x_t} + \mathbf{b_p} \tag{1}$$

where $\mathbf{W_p} \in \mathbb{R}^{d \times m}, \mathbf{b_p} \in \mathbb{R}^d$ are learnable parameters and $\mathbf{u_t} \in \mathbb{R}^d, t = 0, \ldots, w$ are the input vectors of the transformer encoder. To incorporate positional information, the model utilizes a fully learnable positional encoding.

3.2.2. Transformer Encoder. The resulting representation of the input encoding is then fed into self-attention blocks. Each block has two layers. The first is a multi-head self-attention mechanism, and the second is a simple fully connected feed-forward network as proposed in [26]. A residual connection around each of the two sub-layers is applied, followed by batch normalization. Note here that batch normalization is used instead of layer normalization proposed in [26], as batch normalization can mitigate the effect of outlier values in time series, an issue that does not arise in NLP word embeddings [29]. The output of the transformer encoder is the final vector representations $\mathbf{z_t} \in \mathbb{R}^d$ for each time-steps.

3.2.3. Global Temporal Attention. Following the methods presented in [27], the representation $\mathbf{z_t} \in \mathbb{R}^d$ generated from

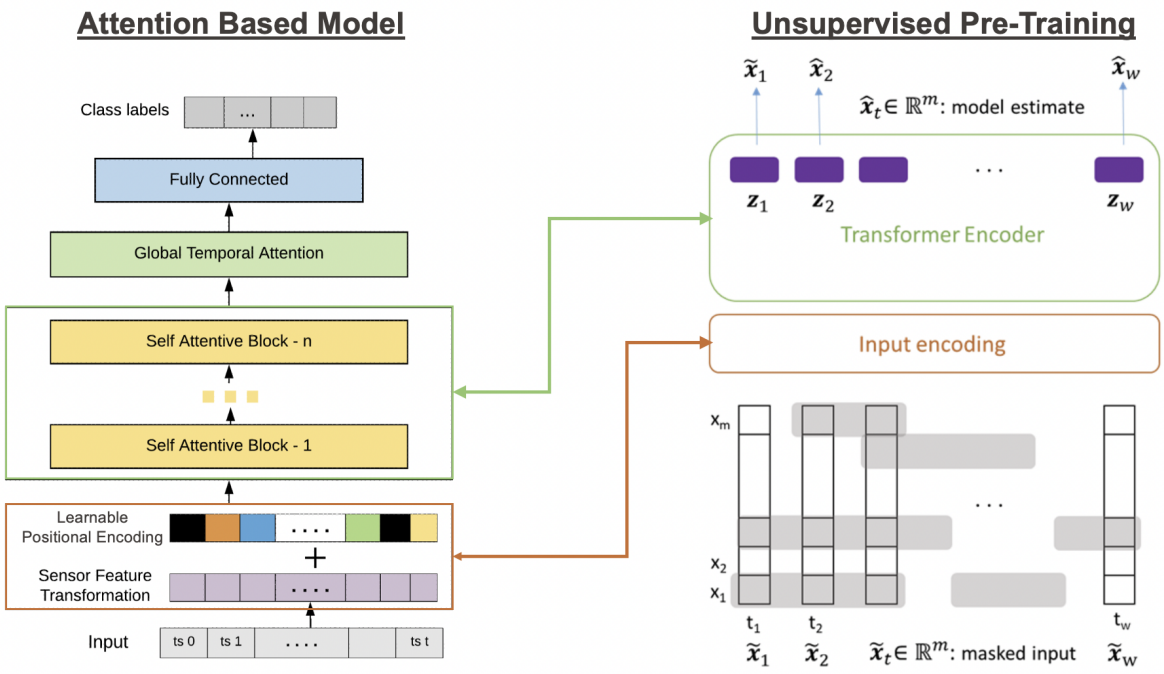

Figure 4. **Left**: Attention based model architecture [27]. **Right**: Training setup of the unsupervised pre-training task [29].

the transformer encoder is utilized by a global temporal attention layer. This layer learns parameters to rank each time-steps according to their respective importance for predicting the corresponding class label for the window. The attention score (ranking) is obtained through equation (3). The terms $\mathbf{W_g}$, $\mathbf{b_g}$ and $\mathbf{g_z}$ are learnable parameters.

$$\mathbf{g_t} = \tanh\left(\mathbf{W_g z_t} + \mathbf{b_g}\right) \tag{2}$$

$$\alpha_t = \frac{\exp\left(\mathbf{g_t}^T \mathbf{g_z}\right)}{\sum_t \exp\left(\mathbf{g_t g_z}\right)} \tag{3}$$

Then, the weighted average, $\mathbf{f} \in \mathbb{R}^d$, of the representations of all the time-steps is computed in equation (4).

$$f^{(i)} = \sum_{t=1}^{w} \alpha_t z_t^{(i)} \quad \text{for i } \in \{1 \ldots d\} \tag{4}$$

Finally, the resulting representation $\mathbf{f} \in \mathbb{R}^d$ is passed through fully connected and softmax layers to obtain a distribution over classes, and its cross-entropy with the categorical ground truth labels is the sample loss to minimize.

3.3. Unsupervised Pre-Training

Transformer-based models are highly expressive and have a large number of parameters, allowing them to capture intricate patterns in the data. However, this high model complexity and capacity can be problematic when data is limited. With fewer examples to learn from, the model may quickly overfit by memorizing the training samples instead of generalizing well to unseen data.

In our case, the limited number of labeled sequences used during supervised learning led to overfitting. To address this, [29] proposed for the first time a transformer-based framework for unsupervised representation learning of multivariate time series. This approach involves pre-training the transformer encoder on unlabeled data to learn meaningful representations, which can then be used for the classification task. By leveraging unsupervised learning, the model can benefit from a larger amount of data and improve generalization performance.

The right part of Fig. 4 shows the training setup of the unsupervised pre-training task. As proposed in [29], a proportion r of each variable sequence in the input is masked independently, such that across each variable, time segments of mean length l_m are masked, each followed by an unmasked segment of mean length $l_u = \frac{1-r}{r} l_m$. Here $l_m = 3$ and $r = 0.15$ as in [29].

A linear layer on top of the final vector representations $\mathbf{z_t}$ is used to make an estimation $\hat{\mathbf{x}}_\mathbf{t}$ of the uncorrupted input vectors $\mathbf{x_t}$:

$$\hat{\mathbf{x}}_\mathbf{t} = \mathbf{W_o z_t} + \mathbf{b_o} \tag{5}$$

Then, only the predictions on the masked values (with indices in the set $M \equiv \{(t,i) : m_{t,i} = 0\}$, where $m_{t,i}$ are the elements of the mask $\mathbf{M}$), are considered in the Mean Squared Error loss for each data sample:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{|M|} \sum_{(t,i) \in M} \sum_{M} (\hat{x}(t,i) - x(t,i))^2 \tag{6}$$

4. Data Collection

4.1. Automatic Labelling

The data collection process with automatic labelling setup is illustrated in Fig. 5.

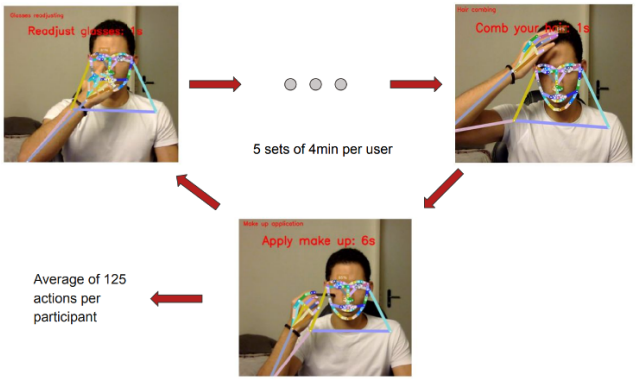

Figure 5. Data collection with automatic labelling setup

The data collection task has been first achieved with an automatic fine grained labeling using a computer vision software, where OpenPifPaf [32] was the main tool for user's motion detection. The software is capable of detecting the user's actions and labeling the data from the wearable device accordingly. The participants for the data collection were to complete a 20 minutes session. The sessions were composed of 5 sets each, with 4 minutes per set.

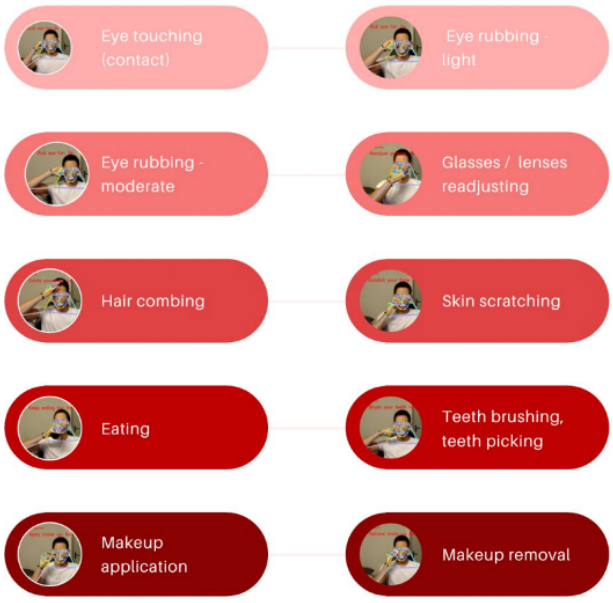

Figure 6. Classes used with the automatic labelling setup

While the computer vision was labeling, the sensor's data of the AppleWatch was recorded by another software called SensorLog [33]. Based on the output of the computer vision software (*user_id*, *class*, *start_time* [timestamp], *end_time* [timestamp]) and the output of the SensorLog application (*timestamp* + 19 features from the sensors of the AppleWatch), each sensor's measure is labeled with the associated *user_id* and *class*. This data collection resulted in signals (time-series of sensor's data with variable lengths) with a label corresponding to one of the classes shown in Fig. 6.

The dataset collected with this setup has constituted a foundational basis for several analyses. Based on the signal statistics, we have set a window size of 3 seconds and a step size of 0.5 seconds to segment the stream of sensor data for real-time prediction. Furthermore, we have found that the 10 classes shown in Fig. 6 are scarcely distinguishable, even for humans. Training any model on this classification task resulted in poor performance. Therefore, we have chosen to group the 10 classes into 4 more meaningful categories (i.e., eye rubbing, face touching, hair combing/skin scratching, and teeth brushing), as depicted in Fig. 2.

However, as this data was recorded in a static position, in an in-vitro setting, the resulting algorithm initially suffered from a high rate of false positives. To overcome this issue, another set of manually labeled data gathered directly from a WatchOS application in real life setting was added. This presented the added benefit of less pre-processing and data cleaning steps as the automatically labelled data.

4.2. Manual Labelling

The data collection process with manual labelling setup is illustrated in Fig. 7.

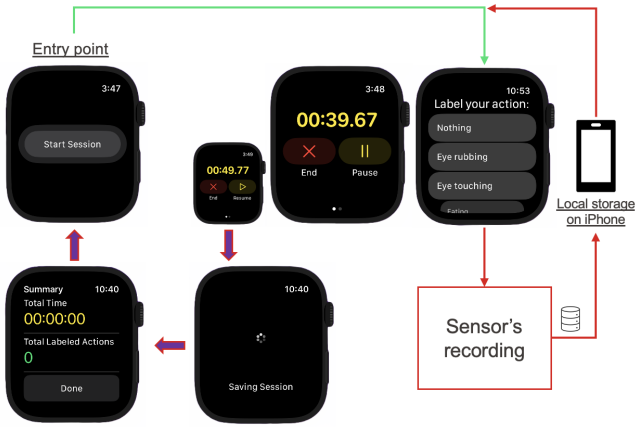

Figure 7. Data collection with manual labelling setup

Several participants were requested to perform hand-face interactions while wearing the AppleWatch. The data collection process was as follow: the participant engaged in the data collection process by explicitly selecting an action from the provided list. Once an action was chosen, the participant had a 2-second window to reach the designated start position. Following the 2-second interval, a haptic feedback and a single ring sound signaled the initiation of the action. The participant started the action at this prompt, and the associated sensor's data was recorded

over the subsequent 3 seconds. Upon completion of the 3-second recording period, two haptic feedbacks with two ring sounds were triggered. It was noted that the actual duration of the participant's action varied; however, what mattered was that the onset of the action fell within the 3-second window. Throughout the data collection, the participant was encouraged to exhibit a diverse range of start positions and perform natural movements.

4.3. Resulting Datasets

Table 1 summarize the statistics for each collected dataset. The automatic labelling setup resulted in signals of variable length. For those signals, we provide statistics of the raw collected signals per user, presented as interactive plots, here [34].

TABLE 1. STATISTICS OF THE COLLECTED DATASETS

	Automatic	Manual	Total
Users	39	11	50
Classes	10 + 1	8 + 1	/
Signals	9531	2000	11531
Total Duration	16h 40min	1h 40min	18h 20min
Eye Rubbing	Light: 612 Moderate: 606	400	1618
Eye Touching	560	100	660
Glasses Readjusting	538	100	638
Eating	535	100	635
Make Up	Application: 282 Removal: 305	100	687
Hair Combing	505	200	705
Skin Scratching	555	200	755
Teeth Brushing	570	400	970
Nothing	4463	400	4863

The resulting dataset collected with the manual labelling setup comprises sequences from 11 users (users 50 to 61). Each user contributes a total of 100 sequences, except for user 50, who contributes 1000 sequences. Each sequences is a signal of 3 seconds of sensor's data. In each user's dataset, an equal number of sequences is allocated for the five classes depicted in Fig. 2. For classes that include sub-classes (e.g., face touching and hair combing/skin scratching), an equal number of sequences is allocated for each sub-class. We publicly share our datasets here.

5. Experiments

5.1. Evaluation Metric

Macro average F1-score is used as the evaluation metric to compare the performance of the proposed approach with other methods. F1-score for each class i is computed as follows:

$$\text{F1-Score}_{\,i} = \frac{2 \times \text{ Precision}_{\,i} \times \text{ Recall}_{\,i}}{\text{Precision}_{\,i} + \text{ Recall}_{\,i}} \tag{7}$$

Then the macro average F1-score is calculated by averaging the statistics for each label:

$$\text{Macro F1-Score} = \frac{1}{|C|} \times \sum_{i=1}^{C} \text{F1-Score}_{\,i} \tag{8}$$

5.2. Train-Validation Split

To ensure an unbiased estimation of the model's performance, the sequences should be split in such a way that each sequence in the training set comes from users who do not have sequences in the validation set. This can be achieved by splitting the sequences based on user IDs. Also, while having overlapping windows in the training samples is not an issue, the validation set should only be populated by non-overlapping sequences.

As previously discussed, the dataset collected using automatic labeling resulted in a high number of false positives after deployment on the watch and required heavy preprocessing to be used for supervised training. Therefore, this dataset was used exclusively for unsupervised pre-training. The collected streams of sensor data were segmented using a sliding window approach, employing a window size of 3 seconds and a step size of 3 seconds for both the training and validation sets (no overlapping). Conversely, the dataset collected using the manual labeling setup was solely used for supervised training. The dataset are split as follow:

- Unsupervised Pre-Training:
  - Train users: 10 to 38
  - Validation users: 40 to 49
- Supervised Training:
  - Train users: 50, 51, 53, 55, 56, 59, 60
  - Validation users: 52, 54, 57, 58

The split for the unsupervised pre-training results in 17238 sequences in the training set and 2774 sequences in the validation set. The split for the supervised training results in 1600 sequences in the training set, with 320 sequences in each of the 5 classes presented in Figure 2, and 400 sequences in the validation set, with 80 sequences per class.

5.3. Effectiveness of unsupervised pre-training

The effectiveness of the unsupervised pre-training method proposed in [29] is evaluated. We trained four versions of the attention-based model both from scratch and using the unsupervised pre-training framework. We used Adam optimizer with cosine warmup scheduler and GELU activation functions.

For the pre-training of the transformer encoder, each versions are pre-trained over 500 epochs using a learning rate of 10^{-3}, 6000 warmup iterations and a batch size of 128. The whole models are then trained for classification over 20 epochs using a learning rate of 5×10^{-4}, 200 warmup iterations and a batch size of 16. For regularization, dropout of 10% and weight decay of 10^{-6} have been used.

TABLE 2. PERFORMANCES OF ATTENTION BASED MODEL TRAINED FROM SCRATCH VS. USING UNSUPERVISED PRE-TRAINING FOR DIFFERENT TRANSFORMER ENCODER CONFIGURATIONS

					Scratch		Pre-Trained	
Version	Nb. Layers	Embed. Size	FC Size	Nb. Heads	Val. Loss	F1	Val. Loss	F1
v0	2	128	512	16	1.32	0.55	**1.16**	**0.57**
v1	4	128	512	4	1.30	0.60	**1.01**	**0.63**
v2	4	128	512	16	1.29	0.58	**1.04**	**0.62**
v3	3	256	256	16	1.33	0.56	**1.05**	**0.60**

Results shown in Table 2 confirm that unsupervised pre-training offers a substantial performance benefit over fully supervised learning both in term of classification performance (F1-Score) and prediction confidence (cross entropy loss).

5.4. Models comparison

We initially established a baseline using traditional machine learning techniques such as K-Nearest Neighbors, Support Vector Machines, and Random Forest. For this purpose, we utilized minimal handcrafted signal features computed in the time domain to adhere to the real-time classification constraint and computational resources available on the Apple Watch. These features include minimum, maximum, mean, standard deviation, skewness, and kurtosis computed for each of the 19 sensor channels. The number of neighbors in KNN is set to 5. In the case of Random Forest, the number of trees is set to 141, and the maximum depth is set to 16.

TABLE 3. COMPARISON OF MODEL'S PERFORMANCES

Model Type	Val. Loss	F1-Score
KNN	/	0.42
SVM	/	0.54
Random Forest	/	0.54
CNN	1.19	0.54
DeepConvLSTM	1.17	0.56
Transformer	**1.01**	**0.63**

We also compared the performances of conventional CNN and DeepConvLSTM models with our best attention based model (Transformer) version. We have implemented the original architecture of DeepConvLSTM presented in [25]. To recall, the DeepConvLSTM architecture consists of four consecutive convolutional layers and two layers of LSTMs. Each convolutional layer is composed with 64 filters, each with a size of 5×1. The convolutions are performed across the time-steps. The output from the last convolutional layer is then passed through a two-layer LSTM, where each LSTM layer has 128 hidden units. The final output vector is connected to a fully connected layer, and the softmax operation is applied to the resulting output. In the fully connected layer, a dropout rate of 50% is applied. The CNN model is obtained by simply removing the LSTM layers from DeepConvLSTM. CNN and DeepConvLSTM models are trained using Adam optimizer with One-Cycle-LR scheduler and ReLU activation functions. Both models are trained over 100 epochs using a learning rate of 5×10^{-4}, a batch size of 16 and a weight decay of 10^{-6}.

Based on the results presented in Table 3, we confirm that the attention-based model (Transformer) outperforms both traditional machine learning and deep learning methods by a significant margin.

5.5. Fine-tuning

The gestures and movements of each individual are subject to individual variations and are not easily generalizable to an algorithm. This explains the poor results achieves on the validation set (only 0.63 of F1-Score). Therefore, we propose a fine-tuning step of the model.

We collected 200 sequences from a new participant (user 62) using the manual data labelling setup. Out of these, 100 sequences were used to fine-tune the attention-based model that showed the best results on the validation set. The performance of the fine-tuned model was then evaluated on the remaining 100 sequences. When fine-tuning the models, we allow training of all the weights.

Additionally, it is notable that half of the sequences in the supervised training set originated from user 50. To gain insights into the model's performance specifically on user 50, we collected an additional 500 sequences exclusively from that user. The evaluation was then conducted solely on these newly collected sequences to assess the model's performance in this specific user context.

6. Results

Performances of the attention based model assessed on the supervised validation set are shown in Fig. 8. Attention based model reached a F1-Score of **0.63**.

Performances of the fine-tuned attention based model assessed on 100 sequences from the new participant are shown in Fig. 9. Attention based model reached a F1-Score of **0.81**.

Performances of the attention based model assessed on 500 sequences from user 50 are shown in Fig. 10. Attention based model reached a F1-Score of **0.95**.

7. Discussion

In this work we first proposed a WatchOS app and a data collection procedure that ensured the capture of genuine hand-face interactions in real-world scenarios. Several other studies have managed to report good results using more

Figure 8. Performances of the attention based model assessed on the validation set (5 individuals, 500 sequences, 100 sequences per class). Attention based model reached a F1-Score of **0.63**

Figure 9. Performances of the fine-tuned attention based model assessed on 100 sequences from the new participant, with 20 sequences per class. Attention based model reached a F1-Score of **0.81**

cumbersome devices or tailored and limited gestures [12], [13], [14], [15]. The current method improves on these and adds the challenge of achieving similar results solely with a wrist-worn device.

Existing models for human activity recognition from sensor reading, whether they are recurrent, convolutional, or hybrid, face challenges in capturing the spatio-temporal context information from the sequences. While CNNs excel at capturing spatial information, LSTM networks were typically required to capture temporal information. However, the Transformer architecture presents numerous advantages over LSTM networks, such as parallel computation, efficient capture of long-range dependencies with attention mechanisms, and mitigation of sequential bias. Transformers are also memory-efficient, scalable for larger sequences, and offer interpretability. In this context, Mahmud and al. [27] came up with a self-attention based neural network model

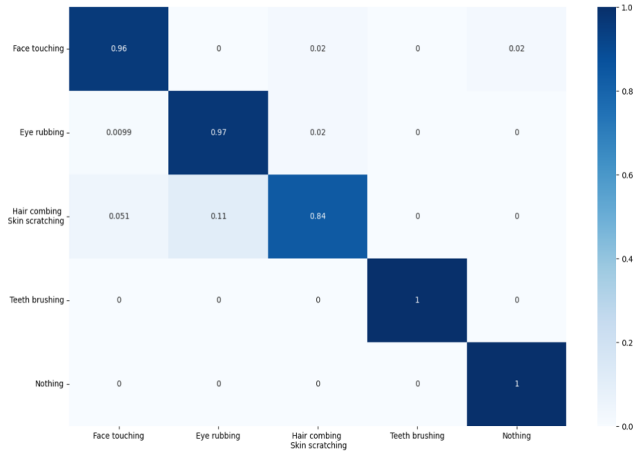

Figure 10. Performances of the attention based model assessed on 500 sequences from user 50, with 100 sequences per class. Attention based model reached a F1-Score of **0.95**

that foregoes recurrent architectures and utilizes different types of attention mechanisms to generate higher dimensional feature representation used for classification. They performed extensive experiments on four popular publicly available datasets: PAMAP2, Opportunity, Skoda and USC-HAD and achieved significant performance improvement over recent state-of-the-art models. Moreover, Zerveas et al. [29] demonstrated the successful application of unsupervised pre-training techniques in the realm of multivariate time series classification. By utilizing extensive unlabeled data, these techniques empower the model to acquire meaningful representations and features that can be further refined for specific classification tasks.

In our study, we demonstrated the substantial superiority of the attention-based model (Transformer) over CNN and DeepConvLSTM in accurately classifying specific hand-face interactions using smartwatch sensors. These findings provide strong evidence for the applicability and effectiveness of Transformer models with self-attention in tackling this challenging task. Additionally, our study confirmed that unsupervised pre-training yielded substantial performance improvements compared to fully supervised learning for this particular task.

The strength of this study is that it managed to successfully achieve promising results in predicting and differentiating among hand-face interactions, and detecting eye rubbing in a real life scenario, with the self-imposed challenges and technological restraints of a solely wrist-worn device. Furthermore, the current algorithm can be further improved by a proposed fine-tuning step based of either 100 sequences (approximately 20 minutes of data collection) or 1000 sequences (approximately 3 hours) provided by the user. Currently, this step is not automated but could be further improved and facilitated in the near future by constant feedback and fine-tuning from the device while being worn, day after day.

8. Conclusion

In the current state, the trained model enables the detection of eye rubbing at 64%, which increases to 80% and 97% with 100 and 1000 sequences respectively. These are commendable results, demonstrating the feasibility of the project, especially with the self-imposed technological restraints, but further improvement is still required in terms of data collection and algorithm optimization. The current need of a 3-hour-long fine-tuning step on a user to achieve good results seems negligible once the algorithm is constantly active and worn, and can benefit from constant feed-back and real-time data collection.

References

[1] A. Mazharian, R. Flamant, S. Elahi, C. Panthier, R. Rampat, and D. Gatinel, "Medium to long term follow up study of the efficacy of cessation of eye-rubbing to halt progression of keratoconus," *Frontiers in Medicine*, vol. 10, 05 2023.

[2] Y. Kwok, J. Gralton, and M. Mclaws, "Face touching: A frequent habit that has implications for hand hygiene," *American journal of infection control*, vol. 43, pp. 112–4, 02 2015.

[3] A. Chen and Y. Li, "Bootstrapping user-defined body tapping recognition with offline-learned probabilistic representation," 10 2016, pp. 359–364.

[4] A. Chen, N. Marquardt, A. Tang, S. Boring, and S. Greenberg, "Extending a mobile device's interaction space through body-centric interaction," 09 2012, pp. 151–160.

[5] V. Vechev, A. Dancu, S. T. Perrault, Q. Roy, M. Fjeld, and S. Zhao, "Movespace: On-body athletic interaction for running and cycling," in *Proceedings of the 2018 International Conference on Advanced Visual Interfaces*, ser. AVI '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3206505.3206527

[6] M. Dias, S. Gibet, M. Wanderley, and R. Bastos, *Gesture-Based Human-Computer Interaction and Simulation, Proceedings of Gesture Workshop 2007*, 12 2009, vol. 5085.

[7] X. A. Chen, J. Schwarz, C. Harrison, J. Mankoff, and S. Hudson, "Around-body interaction: Sensing & interaction techniques for proprioception-enhanced input with mobile devices," in *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices & Services*, ser. MobileHCI '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 287–290. [Online]. Available: https://doi.org/10.1145/2628363.2628402

[8] X. A. Chen, "Faceoff: Detecting face touching with a wrist-worn accelerometer," 2020.

[9] Z. Yang, C. Yu, F. Zheng, and Y. Shi, "Proxitalk: Activate speech input by bringing smartphone to the mouth," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, pp. 1–25, 09 2019.

[10] Y. Dong, J. Scisco, M. Wilson, E. Muth, and A. Hoover, "Detecting periods of eating during free-living by tracking wrist motion," *IEEE journal of biomedical and health informatics*, vol. 18, 09 2013.

[11] J. J. Son, J. C. Clucas, C. White, A. Krishnakumar, J. T. Vogelstein, M. P. Milham, and A. Klein, "Thermal sensors improve wrist-worn position tracking," *bioRxiv*, 2019. [Online]. Available: https://www.biorxiv.org/content/early/2019/02/28/552174

[12] C. Harrison, D. Tan, and D. Morris, "Skinput: Appropriating the body as an input surface," 01 2010, pp. 453–462.

[13] Y. Zhang, J. Zhou, G. Laput, and C. Harrison, "Skintrack: Using the body as an electrical waveguide for continuous finger tracking on the skin," 05 2016, pp. 1491–1503.

[14] X. Zhang, K. Kadimisetty, K. Yin, C. Ruiz, M. Mauk, and C. Liu, "Smart ring: a wearable device for hand hygiene compliance monitoring at the point-of-need," *Microsystem Technologies*, vol. 25, 08 2019.

[15] B. Amento, W. Hill, and L. Terveen, "The sound of one hand: a wrist-mounted bio-acoustic fingertip gesture interface." vol. 724, 01 2002, pp. 724–725.

[16] M. Weigel, T. Lu, G. Bailly, A. Oulasvirta, C. Majidi, and J. Steimle, "Iskin: Flexible, stretchable and visually customizable on-body touch sensors for mobile computing," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, ser. CHI '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 2991–3000. [Online]. Available: https://doi.org/10.1145/2702123.2702391

[17] H.-L. C. Kao, C. Holz, A. Roseway, A. Calvo, and C. Schmandt, "Duoskin: Rapidly prototyping on-skin user interfaces using skin-friendly materials," in *Proceedings of the 2016 ACM International Symposium on Wearable Computers*, ser. ISWC '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 16–23. [Online]. Available: https://doi.org/10.1145/2971763.2971777

[18] I. Poupyrev, N.-W. Gong, S. Fukuhara, M. E. Karagozler, C. Schwesig, and K. E. Robinson, "Project jacquard: Interactive digital textiles at scale," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ser. CHI '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 4216–4227. [Online]. Available: https://doi.org/10.1145/2858036.2858176

[19] S. Ortega-Avila, B. Rakova, S. Sadi, and P. Mistry, "Non-invasive optical detection of hand gestures," in *Proceedings of the 6th Augmented Human International Conference*, ser. AH '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 179–180. [Online]. Available: https://doi.org/10.1145/2735711.2735801

[20] J. Gong, Z. Xu, Q. Guo, T. Seyed, X. A. Chen, X. Bi, and X.-D. Yang, "Wristext: One-handed text entry on smartwatch using wrist gestures," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, ser. CHI '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1–14. [Online]. Available: https://doi.org/10.1145/3173574.3173755

[21] Y. Zhang and C. Harrison, "Tomo: Wearable, low-cost electrical impedance tomography for hand gesture recognition," in *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, ser. UIST '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 167–173. [Online]. Available: https://doi.org/10.1145/2807442.2807480

[22] A. Dementyev and J. A. Paradiso, "Wristflex: Low-power gesture input with wrist-worn pressure sensors," in *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 161–166. [Online]. Available: https://doi.org/10.1145/2642918.2647396

[23] R. Fukui, M. Watanabe, M. Shimosaka, and T. Sato, "Hand-shape classification with a wrist contour sensor: Analyses of feature types, resemblance between subjects, and data variation with pronation angle," *The International Journal of Robotics Research*, vol. 33, no. 4, pp. 658–671, 2014. [Online]. Available: https://doi.org/10.1177/0278364913507984

[24] R. Fukui, M. Watanabe, T. Gyota, M. Shimosaka, and T. Sato, "Hand shape classification with a wrist contour sensor: Development of a prototype device," in *Proceedings of the 13th International Conference on Ubiquitous Computing*, ser. UbiComp '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 311–314. [Online]. Available: https://doi.org/10.1145/2030112.2030154

[25] F. J. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, 2016. [Online]. Available: https://www.mdpi.com/1424-8220/16/1/115

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[27] S. Mahmud, M. T. H. Tonmoy, K. K. Bhaumik, A. K. M. M. Rahman, M. A. Amin, M. Shoyaib, M. A. H. Khan, and A. A. Ali, "Human activity recognition from wearable sensor data using self-attention," *CoRR*, vol. abs/2003.09018, 2020. [Online]. Available: https://arxiv.org/abs/2003.09018

[28] I. Luptáková, M. Kubovčík, and J. Pospíchal, "Wearable sensor-based human activity recognition with transformer model," *Sensors*, vol. 22, p. 1911, 03 2022.

[29] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, "A transformer-based framework for multivariate time series representation learning," 2020.

[30] Apple, "Getting raw accelerometer events." [Online]. Available: https://developer.apple.com/documentation/coremotion/getting_raw_accelerometer_events

[31] ——, "Getting processed device-motion data." [Online]. Available: https://developer.apple.com/documentation/coremotion/getting_processed_device-motion_data

[32] VITA-Lab, EPFL, "Official implementation of "openpifpaf: Composite fields for semantic keypoint detection and spatio-temporal association" in pytorch." [Online]. Available: https://github.com/vita-epfl/openpifpaf

[33] D. B. Thomas, "Sensorlog application." [Online]. Available: http://sensorlog.berndthomas.net/

[34] T. Mery, "Interactive data visualization tool." [Online]. Available: https://temryl.github.io/HFI_DataVisualization/