International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Detecting Face-Touch Hand Moves Using Smartwatch Inertial Sensors and Convolutional Neural Networks

Abdullah Alesmaeil[1], Eftal Sehirli[*2]

Submitted: 02/09/2021 Accepted : 10/02/2022

Abstract: As per World Health Organization (WHO), avoiding touching the face when people are in public or crowded places is an effective way to prevent respiratory viral infections. This recommendation has become more crucial with the current health crisis and the worldwide spread of COVID-19 pandemic. However, most face touches are done unconsciously, that is why it is difficult for people to monitor their hand moves and try to avoid touching the face all the time. Hand-worn wearable devices like smartwatches are equipped with multiple sensors that can be utilized to track hand moves automatically. This work proposes a smartwatch application that uses small, efficient, and end-to-end Convolutional Neural Networks (CNN) models to classify hand motion and identify Face-Touch moves. To train the models, a large dataset is collected for both left and right hands with over 28k training samples that represents multiple hand motion types, body positions, and hand orientations. The app provides real-time feedback and alerts the user with vibration and sound whenever attempting to touch the face. Achieved results show state of the art face-touch accuracy with average recall, precision, and F1-Score of 96.75%, 95.1%, 95.85% respectively, with low False Positives Rate (FPR) as 0.04%. By using efficient configurations and small models, the app achieves high efficiency and can run for long hours without significant impact on battery which makes it applicable on most off-the-shelf smartwatches.

Keywords: *COVID-19, CNN, Hand Activity Recognition, HRT, IMU, Motion Sensors, Sensor Fusion, Wearables*

1. Introduction

As per World Health Organization (WHO), the SARS-CoV-2 virus which causes the COVID-19 infection, can be transmitted between people in two major ways. The first one is through respiratory droplets when a person stays in close contact with an infected person. Or, when people touch a surface contaminated with the virus, then touch their faces specially their eyes, noses, or mouths. To help in lowering infection rates, many recommendations were issued by national health organizations for the people to avoid face touches as much as possible. Many studies show that people tend to touch their faces more than 20 times per hour on average [1], many of these touches are done unconsciously. This makes it difficult for people to monitor their hand moves all the time by themselves in order to avoid undesired face touches, which opens the need for automatic detection and alert solution. In recent years, smart wearable devices like smartwatches have become more popular and widely used by people to monitor their health and fitness. Most of these devices are equipped with Inertial Measurement Unit (IMU) motion sensors like Accelerometer and Gyroscope, in addition to other non-inertial sensors like Magnetometer, Barometer, etc. These sensors can be used to classify hand moves and activities which have many applications like Human Computer Interaction (HCI) [2, 3, 4], sign-language recognition [5, 6, 7], sport actions monitoring [8, 9, 10, 11], or for monitoring daily-life actions [12, 13].

Smartwatches and wearable sensors were used previously in literature to classify hand gestures and actions. Authors in [14] tried to identify 7 different gestures done by the hand and 95.5% accuracy was achieved using 10-Fold cross-validation. Authors in [15] used wearable motion sensors and transferred their data to a connected server for processing, then used Support Vector Machines (SVM) and Artificial Neural Networks (ANN) for classifying 6 different hand gestures. Authors in [16] used wrist-worn sensors plus a Respiratory Inductance Plethysmography (RIP) sensor to classify sensors data for detecting smoking episodes and puffs. For classification, they used Long-Short Term Memory (LSTM) layer on top of Convolutional Neural Networks (CNN) layers. The system in [17] takes sensors readings from a combination of smartwatch and smartphone and used Variational Auto Encoder (VAE) with a neural decision forest to detect smoking events. Authors in [18] used a combination of smartwatch and a magnetic toothbrush to monitor toothbrushing gestures and successfully reached 85.6% average precision.

Face-Touch detection is a new domain and has received focus recently due to COVID-19 pandemic. Detecting face touch hand moves is more challenging because such moves are short with less distinctive features. In addition, they require differentiation between several normal daily-life hand moves like touching body parts, picking items, moving objects…etc. Using extra hardware besides wearable motion sensors can help to overcome some of the challenges [19, 20, 21]. Specifically, the authors in [19] used passive high-functional Radio Frequency Identification (RFID) tags attached to goggles or earrings, with magnetic ring worn on

[1] *Computer Engineering, Karabuk University, Karabük – 78050, TURKEY*
ORCID ID: 0000-0002-8592-6149
[2] *Medical Engineering, Karabuk University, Karabük – 78050, TURKEY*
ORCID ID: 0000-0003-0511-1933
** Corresponding Author Email: eftalsehirli@karabuk.edu.tr*

the hand, then they used K-Nearest Neighbor (KNN) classifier to classify magnetic proximity readings. Authors in [20] used depth sensors on Kinect-like device or special depth cameras to monitor workers behavior and detect when they are moving their hands towards any of the face parts. Authors in [21] used a wrist-worn smartwatch along with a magnetic necklace worn on the neck. By reading the magnetic values, they were able to estimate the proximity between the hand and the face to detect face touch attempts. Using only a smartwatch without extra hardware is a more convenient and cheaper solution. In the study [22], only smartwatch sensors were used to extract features and feed them to a Random Forest (RF) classifier. Authors in [23] took data readings from a smartwatch and compared traditional classifiers like SVM and RF with CNN networks and found CNN outperformed other classifiers.

This work provides a smartwatch application that takes Accelerometer and Gyroscope data and feeds them to small, efficient, end-to-end CNN models to classify hand motion and identify Face-Touch moves. It provides real-time feedback and alerts the user with vibration and sound when attempting to touch the face. The app runs completely on the watch without requiring server calls, or the paired phone to be nearby, and it does not require any extra hardware or calibration. The goal of the app is not just alerting the user when trying to touch the face, but to train user unconscious mind using haptic feedback similar to Habit Reversal Therapy (HRT) [24]. This will assist to develop a habit of avoiding unnecessary face touches even when the app is not running, which in turn will lead to fewer infection rates.

The contributions of this work are:

1. It provides complete and applied solution that runs completely on a smartwatch and uses only IMU motion sensors without requiring extra sensors, devices, or hardware components. It reports state of the art accuracy results on a Test Dataset that mimics real world usage, with recall and precision of 96.75%, 95.1% respectively, and False Positives Rate (FPR) of 0.04%.
2. Instead of approaching the problem as binary classification (Touch/No-Touch), it proposes an alternate approach by dividing hand motion into 5 classes instead of two which was proved to have significant impact on improving precision and minimizing false positives.
3. The use of controlled data collection sessions and automatic labeling algorithm based on peak-valley analysis which allowed to collect large training dataset with 14k samples for each hand, which to the best of our knowledge is the largest dataset for face-touch detection.
4. This work is designed”” from the start to address and solve battery consumption limitations on a small device like smartwatch especially when running the app continuously for long hours. It used the lowest configurations compared with literature from small window size of 0.6s or 30 frames, to extremely compact CNN model with only 11.4k parameters. Testing results on real-world usage show that the app consumes less than 2% of the battery per hour even when running on an old watch.

2. Material and Methods

Two different datasets were collected for left and right hands with the exact methodology. Data are collected using 3 different volunteers with variable age and body characteristics while wearing different watch models with different capabilities. The methodology used for training dataset gathering and processing has a huge impact on achieving both high accuracy and high efficiency.

2.1. Sensors and Configurations

To reduce battery consumption, only Accelerometer and Gyroscope inertial sensors were used. The use of Magnetometer sensor was dropped because it is not supported on all smartwatch models, it has higher power consumption [25], and can be affected by watch metal frame or metal bands, thus, requiring manual calibration to be done by the user. All sensor readings are relative to a local moving reference frame pinned to the device screen as shown in Fig.1. Sliding-Window method is used with window-size set to a small number of 30 frames or 0.6 seconds in order to reduce the calculations required in each model call and save on power consumption. During runtime inference, 50% overlapping is applied which results in 3.33 model-calls per second.

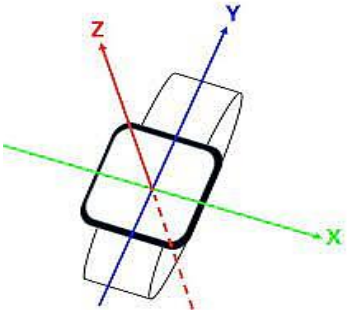
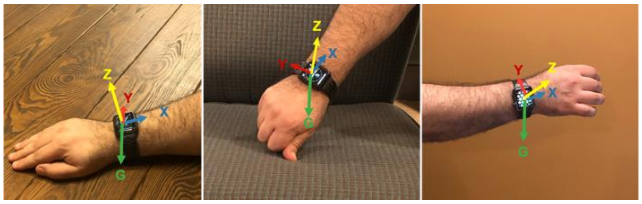


Fig. 1. Local reference frame pinned to watch screen with X-axis goes from left to right, Y-axis top down, and Z-axis is virtual line penetrates the screen bottom up.

Instead of reading raw sensors data, Sensor-Fusion between Accelerometer and Gyroscope was used at low sampling frequency of 50Hz. Sensor-Fusion achieves two main goals: the first one is correcting Gyroscope data by removing accumulated bias. The second is separating linear device acceleration from acceleration due to earth gravity. Input data contain 9 features represent real device linear acceleration, corrected rotation rate, and current gravity vector along 3 axes X, Y, Z. Rotation rate is measured in radians/second whereas both device acceleration and gravity are measured in g (g = 9.81/s2). Gravity can capture device tilt because data are relative to device’s reference frame, and current gravity vector is always vertical and aligned with earth gravity vector,



which means gravity vector angles will change with the device or hand tilt (shown in Fig.2).

Fig. 2. Shows Gravity vector (G) relative to current device’s reference frame represented in 3 axes (X, Y, Z).

Whereas most hand moves start with similar motion patterns, the face-touch move specifically contains some unique hand-tilt patterns depending on which hand-part is touching the face and which face-part is being touched. These tilting patterns provide distinctive features that help the classifier recognize the face-touch move. That is why adding gravity features significantly improved face-touch detection rate or recall as shown in Fig.10.

2.2. Improving Precision by Dividing Hand Motion into Multiple Classes

Hand motion in 3D space is chaotic and some hand moves share many similar acceleration and rotation patterns with the face-touch move. In such binary classification problem (touch/no touch), we were still able to achieve high recall, but precision was relatively low due to many false positives. By closely examining false positives, it was found that they were mostly caused by specific types of hand moves that are elbow-based. This happens when the forearm moves around the elbow making a sharp angle with the upper arm as in Fig.3 (a). This is in contrast to moving the whole arm together or when the angle between the forearm and upper arm is large as shown in Fig.3 (b).

Fig. 3. a) When the forearm moves around the elbow it forms a sharp angle with the upper arm causing spike in rotation values. b) the angle between forearm and upper arm is large.

To minimize false positives, besides "Face-Touch" class, additional two elbow-based motion classes were added. "Up" class represents when the hand moves up with close level to the head, and "Abdominal" class which represents all hand moves towards the abdominal area like stomach and chest. Also, another two classes were added, "Stationary" class for when the hand is stationary or slightly moving, and a final "Normal" class for all other hand moves which brings the total of hand motion classes to five.

Moreover, to achieve fine-grained division and make the classifiers more robust, for each of those classes, data are collected in 5 scenarios: standing, walking, and 3 sitting positions: sitting while hand in a neutral or lower position, or in a medium position like resting on the legs or couch, or in a high position like resting on a table as shown in Fig. 4 (a). The three sitting positions serve another purpose which is recognizing face touch move even if it was initiated from different locations rather than just the neutral position even in standing or walking scenarios as in Fig.4 (b).

Fig. 4. a) 5 different body positions: standing, walking, sitting with neutral hand position, sitting with hand on legs or couch, sitting and hand is on a table. b) initiating hand move from 3 different heights.

Additionally, since local moving reference-frame is used instead of external fixed one, any change in hand orientation or any small shift in watch position around the wrist caused by loose bands will give different data patterns. To overcome this problem, for each of the above 5 scenarios, data are collected with multiple hand orientations. We define 2 hand orientations at the move start, and 3 at the move end that represents 3 different ways for touching the face either with hand palm, back, or side. which makes the total 6 combinations of hand orientations as shown in Fig. 5. Those hand orientations are not rigid, a range of hand rotation in either direction is allowed during samples collection to mimic real-life cases especially with loose bands where the watch can move around the wrist.

Fig. 5. Two hand orientations at the motion start and three for motion end when touching the face for a total of 6 combinations.

Any change in hand orientation at the start may cause different motion patterns like acceleration or rotation on a different axis. Similarly, touching the face with different orientation or different hand-part will cause different rotational patterns. Figure 6 shows different acceleration and rotation patterns for the same hand move in the same standing position, but in 6 different hand orientation combinations.

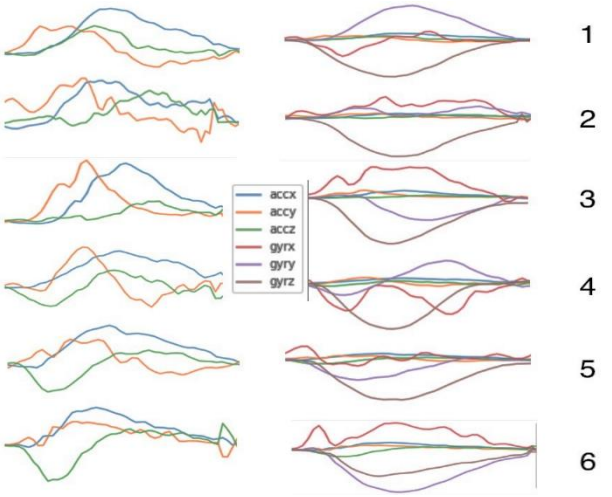


Fig. 6. Sample motion patterns for 6 hand orientations combinations of "Touch" move in standing position. First column represents acceleration values on 3 axes (accx, accy, accz), whereas second column is for rotation rate values (gyrx, gyry, gyrz).

By applying the 3-level division of hand motion: using 5 motion classes, and for each class, collecting data in 5 scenarios, and finally, for each scenario, applying 6 different hand orientations, we allow the models to train on a wide range of motion patterns in real-life situations which improved detection accuracy and helped to minimize false positives.

2.3. Automatic Sample Extraction and Labelling

Training data are collected in controlled sessions where each session is performed by one person for a specific motion class, and only for one scenario, and hand orientation. That means up to 30 different sessions for each motion class. In each session, the volunteer will do repetitive hand moves with around 1 second rest in between. Each session contains between 40 and 150 samples and data are recorded using a special app, where the user will press

start-button to start the session and only presses the stop-button when the whole session is finished.

A special algorithm was developed to extract training samples automatically which helped the collection of large number of samples in less time. Visualizing the 3 elbow-based hand moves (Face-Touch, Up, Abdominal), it was observed that they cause a spike in rotation rate around Z or Y axes resulted from moving the forearm around the elbow. Using this prominent feature besides the fact that there is only one targeted move type in each session, an algorithm that uses peak-valley analysis was developed to extract the samples automatically. Each move starts with an acceleration phase until it peaks at the middle, then a deceleration phase until the hand stops. Sample is extracted by taking the peak value as the middle point (shown in Fig.7).

A small window size of 0.6 seconds was chosen to minimize required calculations and save on power consumption and allow early touch detection. But this is mostly smaller than the actual touch move duration as observed in the dataset which mostly falls in the range of [0.55, 0.8] seconds. Random shift of the middle point was applied with up to 2 frames to the left, and up to 5 frames to the right, with more emphasis on right-shift because the end of touch move contains more distinctive features compared with the move-start. The formula for the extracted-sample-range at detected peak at moment (t) is given in (1) where r is a random number in range of [-2, +5]:

$$\boldsymbol{SampleRange} = \left[\boldsymbol{t} - \left(\frac{\boldsymbol{window_{size}}}{2}\right) + \boldsymbol{r},\ \boldsymbol{t} + \left(\frac{\boldsymbol{window_{size}}}{2}\right) + \boldsymbol{r}\right] \qquad (1)$$

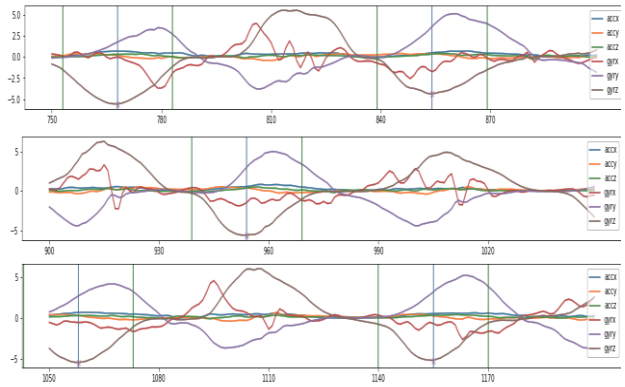


Fig. 7. Example of samples automatically detected by the algorithm. The blue line represents the negative peak value of rotation rate on Z-axis (gyrz) and green lines are the sample start/end boundaries.

Automatic sample extraction algorithm achieves 99% accuracy which means for each 100 samples it correctly extracts 99 samples. Using controlled setup where each volunteer made 80+ separate data collection sessions for each dataset and using automatic sample extraction, we collected a large dataset with around 2800 training samples per class per dataset for a total of 28,000 samples for both left and right hands.

2.4. CNN Architecture

The proposed architecture uses end-to-end models based on CNN networks without any hand-crafted or computed features. Instead of 2D Convolutions, 1D convolutions were used to convolve data along time axis with features set as depth channels. Average pooling layers were used for down sampling. CNN blocks automatically extract 160 features before finally feeding them to dense layers for classification.

Model efficiency is critical, it must have faster execution time and minimum power and CPU usage. By optimizing CNN layers, the final model was reduced to a compact size with only 11.4k parameters. All this was done by reducing the number of convolution layers to 3, reducing filters count in each layer to 32, and having small size of 40 for the first dense layer as shown in Table 1.

Table 1. CNN network structure.

Layer	Parameters
Input	Shape = (30 x 9)
1D Convolution	No. of Filters = 32, Filter Size = 2
Average Pooling	Size = 2
1D Convolution	No. of Filters = 32, Filter Size = 2
Average Pooling	Size = 2
1D Convolution	No. of Filters = 32, Filter Size = 2
Dropout	Dropout Rate = 0.5
Fully Connected	Size = 40
Dropout	Dropout Rate = 0.2
Fully Connected	Size = 5, with Softmax activation

The model consists of 3 convolutional blocks, 2 pooling layers, and 2 dense layers as shown in Fig.8.

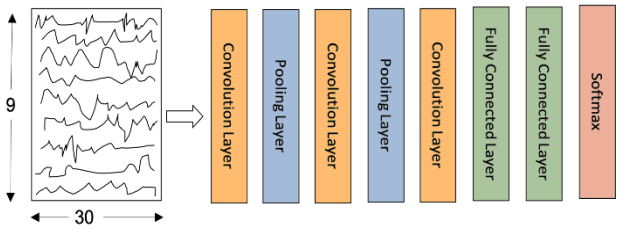


Fig. 8. CNN network input shape and layers.

To find the best hyperparameters, a grid-search of 243 iterations where was run. In each iteration a separate model was trained using one combination of hyperparameters such as filters count, dropout rates, optimizer method, batch size, and epochs count as shown in Table 2.

Table 2. Hyperparameters and their tested values where C1, C2, C3 represent convolution layers 1, 2, and 3 respectively. D1, and D2 represent first and second dropout layers respectively.

Hyperparameter	Values
Filters Count	[C1:32, C2:32, C3:32], [C1:64, C2:64, C3:64], [C1:32, C2:64, C3:128]
Dropout Rates	[D1:0.8, D2:0.8], [D1:0.8, D2:0.2], [D1:0.5, D2:0.2]
Optimizer	Adam, RMSProp, SGD
Batch Size	32, 64, 128
Epoch Count	15, 25, 40

Figure 9. shows the training/validation loss and accuracy over multiple training epochs. In addition, it shows that training and validation loss are close which indicates no overfitting on training data is present.

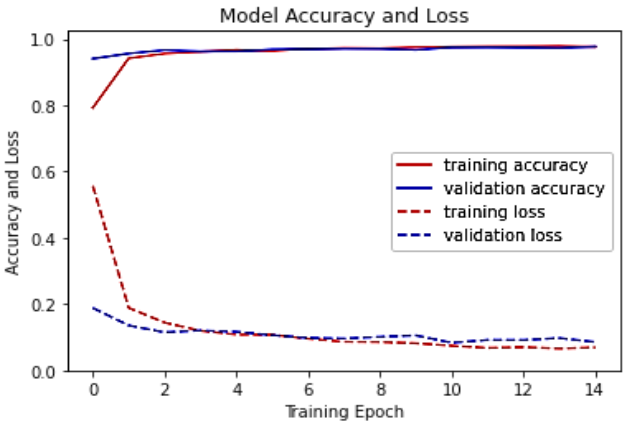


Fig. 9. Training/validation loss and accuracy curves during model training.

Hybrid structure where LSTM layer is added after convolutional layers was tried by some studies [16, 26, 27]. After trying this structure, the model size increased to 32.4k parameters which means more calculations and time are required during inference. Table 3 shows 20.3% increase in inference time on the test dataset when using both LSTM and CNN in hybrid structure. Because model efficiency on a smartwatch is critical, the final model used CNN only model.

Table 3. Comparing CNN only model and CNN+LSTM hybrid model.

	Model Size	Inference Time on Test Dataset
CNN only	11.4k	0.59s
CNN + LSTM	32.4k	0.71s

3. Results and Discussion

3.1. Cross-Validation

Two types of cross-validation were done using the collected left and right training datasets. First, 10-fold cross-validation by splitting training dataset to 10 equal parts with one portion is set as test dataset. Test results show overall accuracy for 5-class classification between 95% and 97% across all 10 folds. The second type of cross-validation is Leave-One-Subject-Out (LOSO) where one subject data is used as test dataset and the rest as training set. Results for all 3 volunteers are close with F1-score between 95% and 96%.

3.2. Separate Test Dataset

To better test the trained models, a special Test-Dataset was collected separately from 3 different volunteers doing real-life activities including face-touches while wearing two different watches. Each person did 50-100 face-touches in the 5 scenarios while doing normal activities for a total of 250 touch samples for each of the left and right hands. Each session is recorded in video and then the dataset is labelled manually by cross-checking all recorded videos. In order to make the test dataset more representative of real-world usage, additional moves that may cause false positives were added, like moving hand up near the face or touching the chest. Those moves are added with 1 to 1 ratio compared with the face-touch move (also around 250 moves per dataset). Additionally, a wide range of normal daily-life activity moves were added like household activities and office activities that mimic the work environment. In total, each dataset contains 250 positive face-touch samples and over 5700 non-touch samples where the hand is doing other motion types, all done over 90 minutes period. Finally, 50% overlapping was applied to match runtime conditions on the app. Test results show state of the art accuracy results with high recall (sensitivity) and precision results with average F1-score of 95.85% (shown in Table 4).

Table 4. Face-Touch accuracy results on the two separate Test Datasets for left and right wrist locations.

	Left Hand	Right Hand	Average
Recall	96.5%	97.0%	96.75%
Precision	93.7%	96.5%	95.10%
F1 Score	95.0%	96.7%	95.85%

As described in section 3.2, adding gravity features increased Recall significantly. By using only linear acceleration and rotation rate features, the best recall score was 88.5% for F1-score of 86.7% (for left hand). But when 3 gravity features were added, the recall jumped to 96.5% for F1-score of 95% as shown in Fig.10.

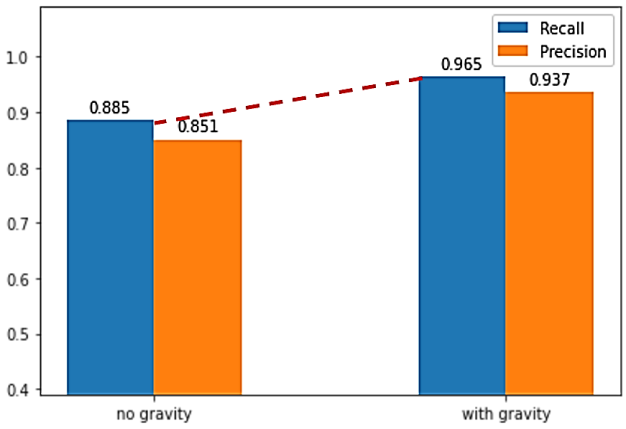


Fig. 10. Recall/Precision results for two models with the best F1-score: first one without gravity features and the second one with gravity included.

Even when testing against over 200 different models trained with different hyperparameters during grid-search. Models with gravity consistently scored higher as shown in Fig.11.

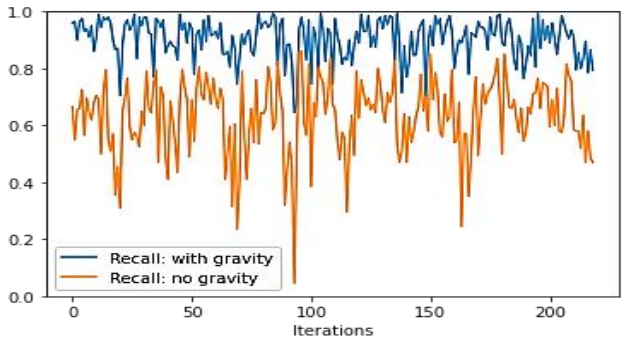


Fig. 11. Recall values with/without gravity for multiple trained models in hundreds of grid-search iterations.

Also, using 5 classes instead of only binary classification had a big effect on minimizing false positives and improving precision. Figure 12 shows large precision improvement for 5-class models with a precision score of 93.7% compared with the best precision of only 75.8% for 2-class models (for models with the best F1-score in both situations).

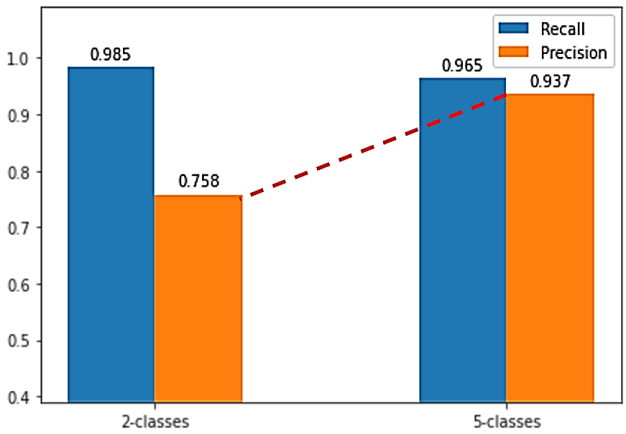


Fig. 12. Recall/Precision scores for two models with the best F1-socre: first one is binary classifier (touch/no touch), and second one classifies 5 different hand motion types.

Again, when testing against multiple models resulted from grid-search iterations, 5-class models consistently scored higher

precision values compared with 2-class models with a big margin as shown in Fig.13. The precision score was mostly in range of 0.5 and 0.7 which is considered very low and not practical for real-world usage on the app even if the recall score is high.

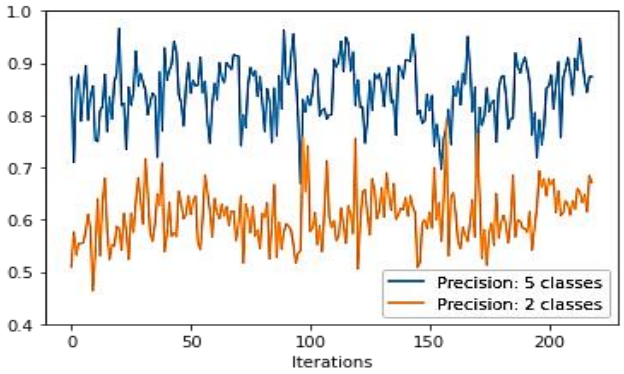


Fig. 13. Shows huge margin in Precision for two classifiers (2-classes and 5-classes) for multiple trained models over hundreds of grid-search iterations.

3.3. Real Tests on the App

To check real-world performance, the final trained models were deployed on the developed application which in turn was deployed on 2 different watches. The app was able to give real-time alert and haptic feedback and show the current count of face touches. 5 persons were asked to use the app extensively and do face touch attempts repeatedly in all 5 scenarios for each hand. The average detection accuracy was 97% for both hands. Additionally, to test false positives rate over long period, each volunteer was asked to use the app for 5 hours period while doing normal daily-life activities and then just report total number of false positives (when the watch alerts the user of a face touch attempt when doing another move). FPR is given in (2) as total False Positives (FP) over the sum of (FP) and all True Negatives (TN):

$$FPR = \frac{FP}{FP+TN} \quad (2)$$

$$Specificity = \frac{TN}{TN+FP} = 1 - FPR \quad (3)$$

for each volunteer data which contain around 30k hand motion samples, the average false positives rate was 0.04% or 1 false positive every 2500 samples which in our setup translates to 2.4 FPs/hour in normal daily life usage. Also, Specificity which is given in (3) has high value of 99.6%. Such very low FPR and high Specificity means the app is alerting the user only when there is very high confidence of potential face-touch move. This makes the app suitable for long hours use without much disturbance for the user that can be caused by too many false alarms.

In terms of battery consumption, the app was power-efficient and battery drain was minimal. Testing results show that running the app continuously in the background added less than 2% battery drain per hour as shown in Table 5. That means even running the app for 10 hours continuously consumes only 20% extra battery. That makes the solution easily applicable in real world where the user can start the app when leaving home and keep it running in the background most of the day without being concerned about battery usage.

Table 5. Battery Usage percentage with and without running the app tested on Apple Watch 3 and SE.

	Battery Level After 5 Hours		
	App is Not Running	**App is Running**	**Consumption by the App**
Apple Watch 3	93%	86%	7%
Apple Watch SE	91%	81%	10%

3.4. Results Comparison

The proposed solution achieved high overall accuracy of 99.7% on the Test Dataset for classifying Touch/No-Touch hand moves while keeping the models power efficient. In Comparison, studies in [19][20] reported accuracy results for touching different face-parts. In [19], authors used RFID tags mounted on plastic goggles in addition to magnetic rings and measured touch detection accuracy for different face locations. Similarly authors in [20] used depth cameras to detect face touches and measure workers compliance with health practices and reported accuracy for touching multiple face parts. All reported accuracy results are shown in Table 6.

Table 6. Comparing accuracy measure of proposed method with other approaches that uses different hardware other than smartwatch.

Study	**Accuracy**
Takayama et al. (2020) [19]	83%
Manghisi et al. (2020) [20]	90.7%
The proposed method	99.7%

The trained models also have a better recall and precision and lower FPR compared with [23] who used feature extraction and CNN networks, and [22] who used hand-crafted features and Random-Forest classifier, and [21] who used magnets to measure the proximity between the hand and the face as shown in Table 7.

Table 7. Comparison for Recall, Precision, and False Positives Rate.

Study	**Recall**	**Precision**	**FP Rate**
Aurizio et al. (2020) [21]	86%	90%	--
Xiang Chen (2020) [22]	89%	--	0.56%
Sudharsan et al. (2020) [23]	91%	--	3.8%
The proposed method	96.75%	95.1%	0.04%

4. Conclusion

In this work, we presented a complete solution for classifying hand moves to detect face touches using only smartwatch IMU motion sensors and CNN networks without using any extra hardware equipment. The goal is to alert users and prevents unwanted face touches which can be one of the main causes for transmitting viral infections. The proposed solution utilizes smart data processing for automatic sample extraction to gather large training dataset with around 28k samples for both left and right datasets. Using sensor fusion helped to eliminate bias and accumulated error and adding gravity features was effective in improving touch detection accuracy. Also, dividing hand motion into multiple classes and collecting data in multiple real-world scenarios and hand orientations led to minimized false positives. This work used small configuration settings compared with literature [28], from small window size to reduced layer count and size. These efficient configurations allowed the models to run directly on the watch with real-time performance while preserving battery at the same time with F1-score of 95.85%.

References

[1] Y. L. A. Kwok, J. Gralton, and M. L. McLaws, "Face touching: A frequent habit that has implications for hand hygiene," *Am. J. Infect. Control*, vol. 43, no. 2, pp. 112–114, 2015, doi: 10.1016/j.ajic.2014.10.015.

[2] D. Moazen, S. A. Sajjadi, and A. Nahapetian, "AirDraw: Leveraging Smart Watch Motion Sensors for Mobile Human Computer Interactions," *arXiv*, pp. 1–5, 2017.

[3] M. C. Kwon, G. Park, and S. Choi, "Smartwatch user interface implementation using CNN-based gesture pattern recognition," *Sensors (Switzerland)*, vol. 18, no. 9, pp. 1–12, 2018, doi: 10.3390/s18092997.

[4] M. Kim, J. Cho, S. Lee, and Y. Jung, "Imu sensor-based hand gesture recognition for human-machine interfaces," *Sensors (Switzerland)*, vol. 19, no. 18, pp. 1–13, 2019, doi: 10.3390/s19183827.

[5] J. Hou *et al.*, "SignSpeaker: A real-time, high-precision smartwatch-based sign language translator," *Proc. Annu. Int. Conf. Mob. Comput. Networking, MOBICOM*, 2019, doi: 10.1145/3300061.3300117.

[6] Q. Zhang, D. Wang, R. Zhao, and Y. Yu, "MyoSign," pp. 650–660, 2019, doi: 10.1145/3301275.3302296.

[7] D. Ekiz *et al.*, "Sign sentence recognition with smart watches," *2017 25th Signal Process. Commun. Appl. Conf. SIU 2017*, 2017, doi: 10.1109/SIU.2017.7960255.

[8] S. Taghavi, F. Davari, H. T. Malazi, and A. Ali Abin, "Tennis stroke detection using inertial data of a smartwatch," *2019 9th Int. Conf. Comput. Knowl. Eng. ICCKE 2019*, no. Iccke, pp. 466–474, 2019, doi: 10.1109/ICCKE48569.2019.8964775.

[9] A. Anand, M. Sharma, R. Srivastava, L. Kaligounder, and D. Prakash, "Wearable motion sensor based analysis of swing sports," 2017, doi: 10.1109/ICMLA.2017.0-149.

[10] S. S. Tabrizi, S. Pashazadeh, and V. Javani, "Comparative Study of Table Tennis Forehand Strokes Classification Using Deep Learning and SVM," *IEEE Sens. J.*, vol. 20, no. 22, pp. 13552–13561, 2020, doi: 10.1109/JSEN.2020.3005443.

[11] G. Brunner, D. Melnyk, B. Sigfússon, and R. Wattenhofer, "Swimming style recognition and lap counting using a smartwatch and deep learning," *Proc. - Int. Symp. Wearable Comput. ISWC*, pp. 23–31, 2019, doi: 10.1145/3341163.3347719.

[12] G. Laput and C. Harrison, "Sensing fine-grained hand activity with smartwatches," *Conf. Hum. Factors Comput. Syst. - Proc.*, 2019, doi: 10.1145/3290605.3300568.

[13] K. Kyritsis, C. Diou, and A. Delopoulos, "End-to-end Learning for Measuring in-meal Eating Behavior from a Smartwatch," *Conf. Proc. ... Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. IEEE Eng. Med. Biol. Soc. Annu. Conf.*, vol. 2018, pp. 5511–5514, 2018, doi: 10.1109/EMBC.2018.8513627.

[14] J. Wu and R. Jafari, "Orientation independent activity/gesture recognition using wearable motion sensors," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1427–1437, 2019, doi: 10.1109/JIOT.2018.2856119.

[15] S. Alavi, D. Arsenault, and A. Whitehead, "Quaternion-based gesture recognition using wirelesswearable motion capture sensors," *Sensors (Switzerland)*, vol. 16, no. 5, 2016, doi: 10.3390/s16050605.

[16] V. Y. Senyurek, M. H. Imtiaz, P. Belsare, S. Tiffany, and E. Sazonov, "A CNN-LSTM neural network for recognition of puffing in smoking episodes using wearable sensors," *Biomed. Eng. Lett.*, vol. 10, no. 2, pp. 195–203, 2020, doi: 10.1007/s13534-020-00147-8.

[17] C. Fan and F. Gao, "A New Approach for Smoking Event Detection Using a Variational Autoencoder and Neural Decision Forest," *IEEE Access*, vol. 8, pp. 120835–120849, 2020, doi: 10.1109/ACCESS.2020.3006163.

[18] H. Huang and S. Lin, "Toothbrushing monitoring using wrist watch," *Proc. 14th ACM Conf. Embed. Networked Sens. Syst. SenSys 2016*, pp. 202–215, 2016, doi: 10.1145/2994551.2994563.

[19] Y. Takayama, Y. Ichikawa, T. Kitagawa, S. Shengmei, B. Shizuki, and S. Takahashi, "Touch Position Detection on the Front of Face Using Passive High-Functional RFID Tag with Magnetic Sensor," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12182 LNCS, pp. 523–531, 2020, doi: 10.1007/978-3-030-49062-1_35.

[20] V. M. Manghisi *et al.*, "A body tracking-based low-cost solution for monitoring workers' hygiene best practices during pandemics," *Sensors (Switzerland)*, vol. 20, no. 21, pp. 1–17, 2020, doi: 10.3390/s20216149.

[21] N. D'Aurizio, T. L. Baldi, G. Paolocci, and D. Prattichizzo, "Preventing Undesired Face-Touches with Wearable Devices and Haptic Feedback," *IEEE Access*, vol. 8, pp. 139033–139043, 2020, doi: 10.1109/ACCESS.2020.3012309.

[22] X. Chen, "FaceOff: Detecting face touching with a wrist-worn accelerometer," *arXiv*, pp. 1–5, 2020.

[23] B. Sudharsan, D. Sundaram, J. G. Breslin, and M. I. Ali, "Avoid Touching Your Face: A Hand-to-face 3D Motion Dataset (COVID-away) and Trained Models for Smartwatches," *ACM Int. Conf. Proceeding Ser.*, no. October, 2020, doi: 10.1145/3423423.3423433.

[24] K. S. Bate, J. M. Malouff, E. T. Thorsteinsson, and N. Bhullar, "The efficacy of habit reversal therapy for tics, habit disorders, and stuttering: A meta-analytic review," *Clin. Psychol. Rev.*, vol. 31, no. 5, pp. 865–871, 2011, doi: 10.1016/j.cpr.2011.03.013.

[25] K. Katevas, H. Haddadi, and L. Tokarchuk, "Sensing Kit: Evaluating the sensor power consumption in iOS devices," *Proc. - 12th Int. Conf. Intell. Environ. IE 2016*, pp. 222–225, 2016, doi: 10.1109/IE.2016.50.

[26] S. Mekruksavanich, A. Jitpattanakul, P. Youplao, and P. Yupapin, "Enhanced hand-oriented activity recognition based on smartwatch sensor data using LSTMs," *Symmetry (Basel).*, vol. 12, no. 9, pp. 1–19, 2020, doi: 10.3390/SYM12091570.

[27] F. J. Ordóñez and D. Roggen, "Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition," *Sensors (Switzerland)*, vol. 16, no. 1, 2016, doi: 10.3390/s16010115.

[28] W. S. Lima, E. Souto, K. El-Khatib, R. Jalali, and J. Gama, "Human activity recognition using inertial sensors in a smartphone: An overview," *Sensors (Switzerland)*, vol. 19, no. 14, pp. 14–16, 2019, doi: 10.3390/s19143213.