# PPGSign: Handwritten Signature Authentication using Wearable PPG Sensor

A B M Mohaimenur Rahman*, Yetong Cao†, Xinliang Wei‡, Pu Wang*, Fan Li†, Yu Wang‡
* Department of Computer Science, University of North Carolina at Charlotte, Charlotte, North Carolina, USA
† School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China
‡ Department of Computer and Information Sciences, Temple University, Philadelphia, Pennsylvania, USA

2022 IEEE Wireless Communications and Networking Conference (WCNC) | 978-1-6654-4266-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/WCNC51071.2022.9771813

***Abstract*—Handwritten signature authentication is a crucial service to defend against fraudulent activities. Existing automated solutions rely heavily on dedicated devices that are expensive and require different user efforts that affect the user experience. In this paper, we propose a new signature authentication system, PPGSign, which leverages Photoplethysmography (PPG) sensors in the existing wrist-worn wearable devices. The unique blood flow changes in the supplicant's hand movement are exploited in this system to validate the signature. To make PPGSign non-intrusive and secure, we explore effective algorithms to separate the signature signals from the heartbeat signals in the raw PPG signals. We build a low-cost hardware prototype to verify our proposed method. Our experimental results show that PPGSign can achieve an average F1 score of up to 98%, which verifies the feasibility and efficiency of the proposed solution.**

## I. Introduction

Paper checks continue to lead the way of transactions at financial institutions and are susceptible to fraudulent attacks. Besides paper checks, other important legal and financial documents still require handwritten signatures to verify a person. Attackers try to forge those signatures in order to bypass the system to conduct their fraudulent activities. To help prevent these fraudulent activities, different automatic handwritten signature authentication systems [1] have been developed. These include offline solutions based on image processing [2], and online solutions based on cameras [3] or touch screen/digital signing pads [4]. Most recently, solutions based on motion sensors on wearable devices [5], [6] and acoustic sensors on mobile devices [7] have also been used for signature authentication. These methods enjoy the low cost, non-intrusiveness, and easy deployment. However, many of them still require users' extra effort (for calibration or training) and suffer from low accuracy (due to environmental noises or attacks). Therefore, there is still a need for new low-cost, non-intrusive, pervasive, robust signature authentication methods.

Photoplethysmography (PPG) sensor nowadays can be found in many wearable devices (e.g. smartwatches and fitness trackers) and has been used to monitor pulse rate and other health aspects [8]. The optical signal reading from PPG sensors can estimate the changes in the blood volume under the skin, thus has been widely used in medical applications. With the increasing popularity of wearables equipped with built-in PPG sensor, PPG-based biometric authentication has emerged as one of the important non-intrusive mobile authentication methods [9], [10], recently.

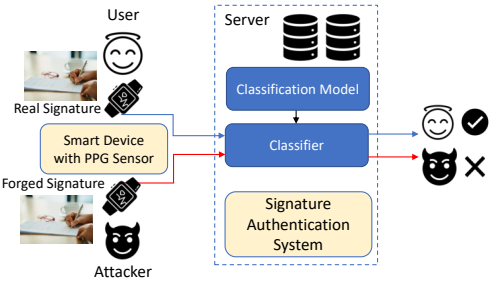


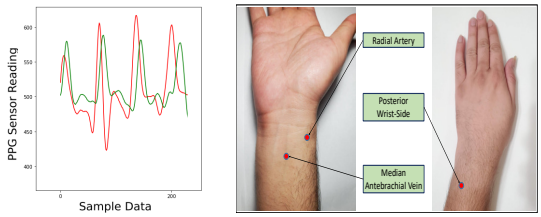

Fig. 1. Handwritten signature authentication via PPG sensor.

Different from user authentication, in this work, we propose a signature authentication system to verify user's handwritten signatures by leveraging the PPG data from a wrist-worn wearable (Fig. 1). Currently, in financial institutions, the user's signatures are either manually verified or automatically verified by an online signature authentication software on a tablet or dedicated smart device. Instead, we exploit PPG sensor from wearable devices to capture the dynamic information of the blood volume change under the skin when a user is writing his signature. Our solution is also different from existing PPG-based user authentication since a handwritten signature is beyond traditional user authentication. We more focus on the dynamic part of PPG-signal caused by the hand/finger movements. The main contributions of this paper include:

- We propose a handwritten signature authentication system (PPGSign) leveraging PPG sensors in wrist-worn wearables. According to our knowledge, PPGSign is the first work that purely uses a PPG sensor for non-intrusive and secure handwritten signature authentication.
- We have explored several segmentation algorithms to separate the signature signals from the raw PPG signal consisting of both heartbeat and signature signals.
- We build a low-cost hardware prototype of PPGSign using a commercially off-the-shelf (COTS) PPG sensor and a micro-controller to evaluate the feasibility of the proposed system. The experimental results show that PPGSign can achieve an average F1 score of up to 98%.

## II. Problem, Model and Challenges

The *signature authentication problem* studied here is primarily a verification problem over a handwriting signature via wearable sensing data. A user will give sensing data of his signature to claim to be a certain person. The system should




(a) PPG signals (b) PPG sensor placement

Fig. 2. Challenges of PPGSign: (a) different PPG pulse signals from the same user; (b) locations for PPG sensor placement.

verify the claim whether the user is a legit or illegitimate user. Usually, a user first registers into the system by providing sensing data of his own signature. The system is trained on that registered signature to create a model based on the features extracted from that signature. As shown in Fig. 1, when a new signature input is given, the system based on the saved models decides whether the user is a legitimate user or an attacker.

The data of the signature can be collected via different sensors, such as using additional hardware (e.g., cameras, signature pads, or custom-built sensors) or leveraging the COTS smart devices (e.g., smartwatches or wristbands). In this work, we leverage a single Photoplethysmography sensor (which is commonly available on most wrist-worn wearable devices) to collect PPG data during the signature. Since the trajectory of the used pen affecting the blood flow of the user's wrist, such an effect is reflected upon the PPG reading samples over time. We use $S(n)$ ($n \in \{1, \cdots, N\}$) to represent the sampled data, and the sampling rate is 100 Hz.

Our system tries to prevent fraudulent activities in scenarios where an illegitimate user is trying to forge a signature to claim to be a different person. The *security model* consists of four main entities, as shown in Fig. 1:

- *Users:* This is the authorized person who should be the only one to be approved by the system. A user registers his signature in the system wearing the smartwatch with a built-in PPG sensor. The system trains on those registered signatures to create a model which later verifies the user.
- *Smart Devices:* Smart devices (such as wrist-worn wearable devices) have a built-in PPG sensor to collect the PPG readings while the user/attacker gives the signature. We assume that they are trustworthy.
- *Server & Links:* The PPG sensor data from the smart device is sent to the server for data processing and verification. We assume that the server itself and the communication link between devices and the server are all trustworthy.
- *Attackers:* An attacker tries to forge the signature of the legitimate user to bypass the system to do fraudulent activities. We assume that the attacker can come close to the user and shoulder peek to observe the signature and the way the user gives his signature.

To successfully validate the fine-grained signatures via the captured PPG signals from a wrist-worn device, there are primarily three challenges to be addressed.

(1) *Coarse-grained wrist PPG signals:* PPG signals are relatively coarse-grained, noisier, and interfere with other signals more than ECG signals [10]. And wrist-worn PPG techniques are even more coarse-grained. The critical landmarks are more detectable in the fingertip region-based PPG signals than in the wrist region ones. This means methods applicable to fingertip PPG will not apply to the wrist PPG. In addition, the signal is generally contaminated with noises due to subtle hardware capture issues. To tackle this challenge, we carefully design our noise filtering to extract critical landmarks (Section III-B) and propose new feature extraction to extract discriminating features (Section III-C) for fine-grained authentication.

(2) *Same user having different PPG readings:* This is also one of the major challenges in existing PPG-based authentication systems [9], [10]. PPG signal contains the pulse signal along with the signature signal from the user. However, the PPG readings are different even for the same user (Fig. 2(a)), as the pulse signal may vary due to the effect of pressure and emotions of the user. Therefore, it is challenging to use the PPG signals for signature authentication directly. To address this, we extract the signature portion signal from the whole signal by a new segmentation method (Section III-B) so that the effect of pressure and emotion is minimized.

(3) *Effect of the placement of the PPG sensor:* The placement of the PPG sensor can affect the performance since PPG readings vary at different locations. As the PPG sensor is generally used for measuring heart rate, blood pressure, and pulse oximetry, existing applications concentrate on the pulse signal measurement from the *radial artery*. Another location for the placement, the most common one for the existing smartwatches/fitness trackers, is the *posterior wrist-side*. In our work, we focus on the minute movements of fingers along with the hand itself while writing the signature. Therefore, we have built our own wrist-worn wearable device in which the PPG sensor is not attached to the Velcro band, and we could test different positions. Via experiments, we find out that if the sensor is placed around the *median antebrachial vein*, the readings would represent the significant motion artifacts occurring due to the signature writing. Though this position is not in line with most people's wearing habits, it is feasible because the user will just have to rotate the wearable device while writing the signature. Also, it is assumed that the user will be wearing the hand-worn device on their dominant hand which would be used to perform the handwritten signatures. See Fig. 2(b) for these three locations. The details of the impact of the sensor placement are discussed in Section IV-B.

## III. Signature Authentication with PPG Sensor

### A. Overall Design

As shown in Fig. 3, PPGSign consists of four parts: *Data Collection*, *Data Processing and Segmentation*, *Feature Characterization*, and *Classification*. In the *data collection* module, the wrist-worn wearable device with a PPG sensor is used to collect the PPG data during the signature written on a sheet of paper. In the *data processing and segmentation* module, there are three steps of data processing: noise filter, signal normalization, and signature segmentation. The PPG signals first pass noise filters and are normalized, then the signature

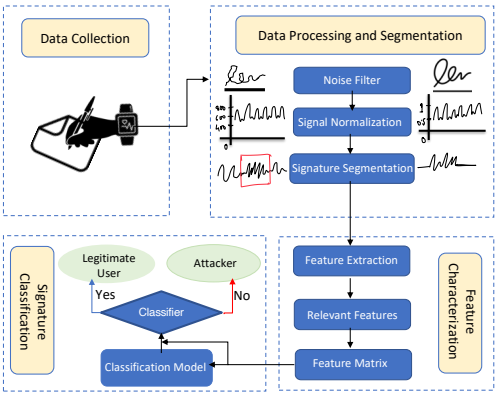



Fig. 3. System architecture of PPGSign.

part of signals are segmented from the whole signals to remove the pulse part. The *feature characterization* module includes feature extraction, relevant features, and feature matrix. This module mainly deals with the extraction of relevant intrinsic characteristics of the input signal data that can discriminate each user from another. Finally, the *classification* module performs the authentication with a trained classifier. The classification model is trained on the whole dataset of a user in order to classify a new incoming signal into any of the two categories: legitimate user or attacker.

During *training phase*, each new user provides some sample signatures via a wrist-worn wearable device. The collected PPG signal is then sent to the server-end and goes through four parts of PPGSign to train a model for each user. During *authentication phase*, the user trying to gain access would provide the signature data, which is sent to the server-end for the same procedures. The classifier compare with the stored pre-trained model and make its authentication decision.

### B. Data Processing and Segmentation

We now introduce the details of data processing and segmentation module, which includes *noise filtering*, *data normalization* and *signature segmentation*, as illustrated in Fig. 3.

*1) Noise Filtering:* Due to the user's behavioral changes and surrounding environmental variations, there are noises in the raw PPG data collected. In addition, there are baseline drifts and high-frequency interference in the PPG readings [10] due to hardware imperfections. Human heart rate is generally around 50-100 beats per minute [9], i.e., its frequency is ranging from 0.8 to 1.7Hz. A naive way is applying a high band-pass filter or Butterworth filter with a high cut-off frequency at 2Hz. However, the signature portion caused by the signature might have some elements with a low frequency that will get filtered by this method. Therefore, instead, we apply a Savitzky-Golay (S-G) filter [11] to smooth the signal. S-G filter is a type of low-pass filter, which is a smoothing method based on local least-squares polynomial approximation.

*2) Data Normalization:* The PPG reading is generally ranging from 400 to 750, and different users have a different variety of data within the range. Before the PPG data is processed for feature characterization, we will normalize it within a range of 0 to 1. Doing so can make the features more consistent and help the convergence during the training phase.

**Algorithm 1** Skewness-DTW Method

Initial $skewList$; $windowSize = 100$; $pulseSize = 80$
**for** $i = 0 \rightarrow (length\ of\ S(n) - windowSize)$ **do**
  $value = SKEWNESS(S[i : (i + windowSize)])$
  Append $value$ to $skewList$
$startPoint = Index\_Min(skewList)$
Initial $scoreList$; $pulseProfile = S[0 : pulseSize]$
**for** $i = startPoint \rightarrow (|S(n)| - pulseSize)$ **do**
  $dataCompare = S[i : (i + pulseSize)]$
  $score = DTW(pulseProfile, dataCompare)$
  Append $score$ to $scoreList$
$endPoint = Index\_Min(scoreList)$
**return** $startPoint,\ endPoint$

**Algorithm 2** DP / Binary Method

Assign $l1$ / $normal$ to Cost Function
Fitting the signal based on the Cost Function
$bkps = DP\_PREDICT()$ / $Binary\_PREDICT()$
$startPoint = bkps[0]$
$endPoint = bkps[-2]$
**return** $startPoint,\ endPoint$

*3) Signature Segmentation:* In this step, the processed data $S(n)$ (still a mixed signal of the user's pulse profile and signature motion) is segmented so that the signature portion is obtained as an output. We explore 3 segmentation algorithms.

Algorithm 1 shows the *Skewness-DTW* method. It first finds out the start index $startPoint$ of the signature portion based on skewness values (calculated by function $SKEWNESS()$) of the signal with a fixed window size $windowSize$. The index of the minimum value of skewness gives us the start point of the signature portion. To find out the end index $endPoint$ of the signature portion, our algorithm uses the Dynamic Time Warping (DTW) technique similar to [12]. In the beginning, it requires a pulse profile $pulseProfile$ of the user, which is obtained from the first few seconds of the signal where the user did not start writing his signature instead keeps his hand/wrist static. A sliding window-based approach is taken where the window size is considered the average size of a single pulse of the user. Each window of the signal after the starting point obtained in the previous step is compared with the pulse profile of the user based on the DTW method (function $DTW()$), and a score is generated, which indicates the similarity between them. Finally, the endpoint is detected based on the minimum value of the DTW scores. Fig. 4(a) shows an example of the output, which marks both the start and end index of the signature portion.

The other two segmentation methods (based on *dynamic programming* (DP) and *binary search* (Binary)) are offline change point detection methods inspired by [13]. Algorithm 2 shows both algorithms (red for DP and blue for Binary). In DP, the costs of all the sub-sequences of the signal are computed

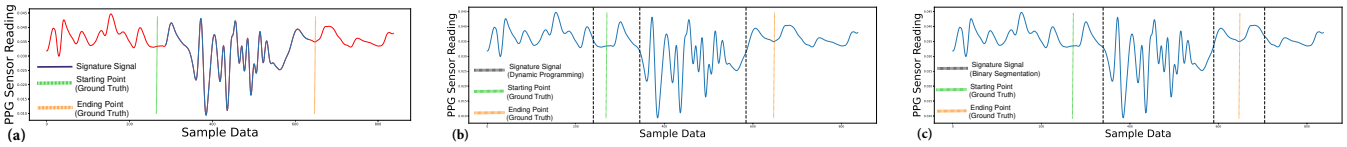



Fig. 4. Examples of different segmentation of signature signal: (a) Skewness-DTW method, (b) dynamic programming, (c) binary segmentation.

at first, and then the minimum of the sum of the costs is calculated. In this process, the number of change points to detect has to be predefined. Via experiments, we let the number of change points be 3, and the first/last change point is the start/end point of the signature signal. $l1$ is a least absolute deviation-based cost function, which is passed as an argument to the cost function model. $DP\_PREDICT()$ predicts the predefined number of breakpoints of the given signal after fitting the signal based on the cost function. Binary is a greedy sequential algorithm that searches for the point for which the sum of the costs gets lowered. After the first change point is detected, the signal is split into two parts on that point. The algorithm is repeated on the sub-signals until the stopping condition is satisfied. $normal$ is a Gaussian Process Change-based cost function. $Binary\_PREDICT()$ predicts the predefined number of breakpoints of the given signal after fitting the signal based on the cost function. This algorithm works without the predefined number of change points, but we use the same number of three change points. Fig. 4(b) and 4(c) show examples of the outputs of these two algorithms.

### C. Feature Characterization

Next, the segmented signature portion is passed on to the feature characterization module, which has 3 following steps.

*1) Feature Extraction:* The PPG features that we focus on are the time-domain features. We use a time-series feature extraction tool *tsfresh*, a python package to calculate time series characteristics or features automatically. With about 63 characterization methods, *tsfresh* by default can generate more than 700 time-series features in an accelerated way. In PPGSign, from each data frame from the clean and segmented signature data, we use *tsfresh* to extract 787 time-domain features, such as the absolute energy, descriptive statistics on the auto-correlation, and binned entropy of the power spectral density of the time series.

*2) Relevant Features:* Next, we also perform feature selection to select relevant features for the time-series classification. It filters the important features for the machine learning model beforehand, which helps to train the model well as there are fewer irrelevant features. Each feature vector generated in the previous step is individually tested to predict the labels in regards to their significance. These tests give a score vector for each feature vector, and these score vectors are evaluated on the basis of the Benjamini-Yekutieli procedure to decide which features to be selected [14]. Out of the generated 787 features, we select 8 features primarily as the relevant features by using *tsfresh*. The selected ones are shown in Table I.

*3) Feature Matrix:* After collecting $m$ calibrating sample signals from a user, the relevant features, say $r$ features, are generated for each sample. Each feature value might have a different range, thus feature scaling is done via $Y_{new} = \frac{Y - mean}{standard\ deviation}$, where $Y$ is the current feature value of the sample, and the new feature value $Y_{new}$ is calculated using the mean and standard deviation of all samples. Now we have a $2D$ matrix of the dimension $m \times r$ where each row represents the signature sample data, and each column represents a selected relevant feature. This matrix is used for the training of the classification model of this user.

TABLE I
LIST OF RELEVANT FEATURES USED BY PPGSIGN.

| Features | Description |
|---|---|
| count_below_mean($x$) | # of values lower than the mean of $x$. |
| number_peaks($x$,$n$) | # of peaks of at least support $n$ in $x$. |
| range_cnt($x$,$min$,$max$) | # of values within [$min$, $max$). |
| fft_coefficient($x$,$par$) | fourier coefficients from FFT. |
| number_cwt_peaks($x$,$n$) | different peaks in $x$. |
| abs_sum_of_changes($x$) | sum over the absolute value of consecutive changes in $x$. |
| agg_linear_trend($x$,$par$) | linear least-squares (LLS) regression for values aggregated over chunks. |
| linear_trend($x$,$par$) | LLS regression for values in $x$. |

### D. Classification

For classification, we use the feature matrix to train the classification model for each user. We use the following different standard classifiers: Random Forest (RF), Support Vector Machine (RBF Kernel), Gradient Boosting (GB), k-nearest neighbor (kNN), Multilayer Perceptron (MLP), Feed-Forward Neural Network (NN), and OnevsRest (OvR). These trained classifiers can determine whether the new input signature PPG data belongs to a legitimate user or not.

## IV. IMPLEMENTATION AND EVALUATION

### A. Prototype and Data Collection

Our prototype of PPGSign system includes two hardware parts: *PPGSign Band* and *Server*. Although the commercially available smartwatches and fitness trackers use PPG sensors to measure the heartbeat/pulse, they do not provide access to the raw PPG data. Thus, we build our own low-cost proof-of-concept PPG-band, *PPGSign Band*, as shown in Fig. 5(a), from the commercially off-the-shelf products. This band aims to imitate wrist-worn wearable devices to validate the feasibility of PPGSign. It consists of a velcro wristband, a PPG sensor by World Famous Electronics with a green LED (as the green LED performs the best), a USB cable (to connect the micro-controller to the server), an Arduino UNO micro-controller, and a slide switch to start and stop collecting data. The PPG sensor is strapped to the Velcro band so that it

remains facing towards the wrist when the Velcro band is worn. A Dell Inspiron 15 laptop is used as the server which is connected to the Arduino with USB cable.

All software modules of PPGSign are implemented in the server using $Python$ 3.8.3. The library package *ruptures* [13] is used for implementation of Algorithm 2. We also use the *tsfresh* tool [15] for feature extraction, and the python package *scikit-learn* [16] for building the classifiers.

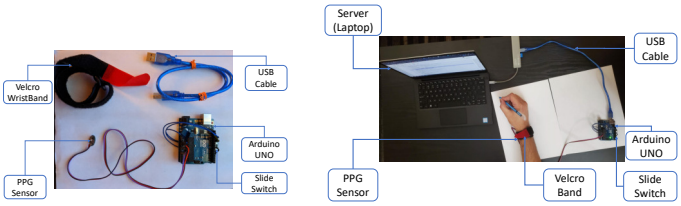
To validate the feasibility of PPGSign, we use the developed PPGSign Band to collect the user's raw PPG data while writing the signature. The user sits on a chair and rests his hand on a piece of paper on which he/she will provide the signature with a pen. The PPGSign band is wrapped around the wrist with the PPG sensor facing towards the Median Antebrachial vein. Fig. 5(b) shows the data collection setup. The PPG data is sampled at a rate of 97.5 Hz from the sensor via the Arduino micro-controller when the side switch is on.

Because of the logistic limitations being created due to the COVID-19 pandemic situation, 5 healthy volunteers participated in the experiment. Participants take part in 6 sessions of signature writing. During each session, the participants provide 20 valid signatures of themselves, and 10 random forgeries and 10 skilled forgeries against each of the other four users. We adopt the same definition of random and skilled forgeries from [5]. In random forgeries, the attacker does not know the legitimate user's signature, while in skilled forgeries the attacker trains on the claimed user's signature. Totally, we have collected $3,000$ samples of signatures. As the COVID-19 situation is relaxing now, we plan to collect more data to expand our findings further.

### B. Evaluation

We have conducted a performance evaluation on our current PPGSign system. For the evaluation, we have tested the data collected via *PPGSign Band* on multiple classifiers with the different splitting of training and test data. User-specific model training was adopted in our system, where a model is trained for each users. Our evaluation mainly focuses on random forgery and skilled forgery attacks on the legitimate user from others users. Besides that, we also test a special case of attacks where the legitimate user fake his "signature". We perform tests on three signature segmentation algorithms from Section III-B and all classifiers mentioned in Section III-D. For splitting training and test data, we use a different portion of samples as the training data, from 20% up to 80%.

Three main metrics are used: (1) *Precision*, the ratio of correctly predicted positive values to the total predicted positive values, i.e., $Precision = \frac{TP}{TP+FP}$; (2) *Recall*, the ratio of correctly predicted positives values to the actual positive values, i.e., $Recall = \frac{TP}{TP+FN}$; (3) *F1 score:* F1 score is the ratio of the positive and negative class, i.e., $F1 = \frac{2 \times Precision \times Recall}{(Precision+Recall)}$. Here, $TP$ represents the number of all $True\ Positive$ cases, i.e. legitimate users are recognized; $FP$ represents the number of $False\ Positive$ cases, i.e. our system approves illegitimate users; and $FN$ represents the number of $False\ Negative$, i.e., our system denies legitimate users. Obviously, we prefer high precision, recall, and F1 score.




(a) PPGSign band (b) data collection with PPGSign

Fig. 5. PPGSign Prototype: (a) hardware components of PPGSign Band; (b) data collection using the PPGSign.

Next, we report the detailed experimental results based on the impacts of different factors, such as types of segmentation methods, classifiers, training sizes, placement of the sensor, attack from same user, and different surfaces. All the results are based on the random and skilled forgery attacks except "attack from same user".

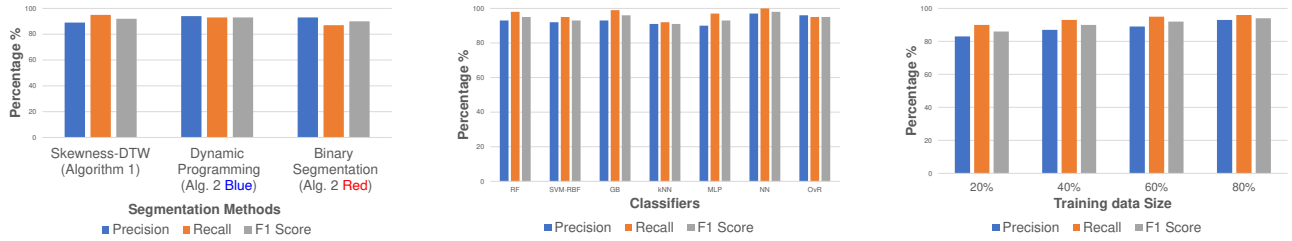
**Impact of Segmentation Methods:** In Section III-B, we introduced three segmentation algorithms (Algorithms 1/2). Fig. 6(a) shows the average performance of PPGSign with all the classifiers when the dataset was split into 60% training and 40% testing. Among the three segmentation methods, Skewness-DTW and DP both performed well. In the rest evaluations, we only report the results where the Skewness-DTW method is used due to space limitations.

**Impact of Classifiers:** We compare 7 commonly used classifiers as listed in Section III-D. The classifiers are considered under the Skewness-DTW segmentation method and with a training size of 80%. The F1 scores for each classifier are shown in Fig. 6(b). Feed-Forward Neural Network performs the best with the highest F1-score of 98%, while the scores of RF/GB/OvR are also within a close range.

**Impact of Training Sizes:** We then choose different percentages of data to use as the training data. Fig. 6(c) shows the average performance of Skewness-DTW segmentation method and all classifiers when training data size is from 20% to 80%. Overall, when the size of the training dataset is increased, the average F1-score also increases from 86% to 94%. This basically shows that our system can take just a few training data to achieve reasonable accuracy.

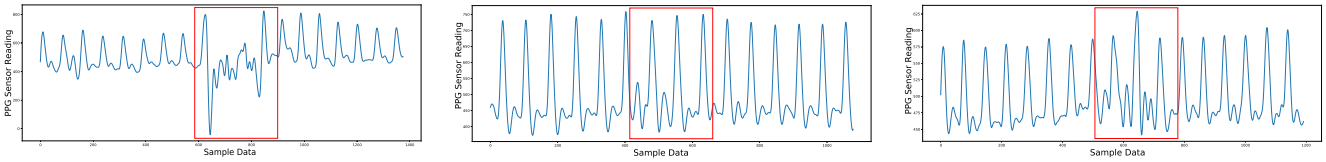
**Impact of Placement of Sensor:** As mentioned in Section II, the placement of the sensor within the wrist area is critical. We test three different sensor placements shown in Fig. 2. Fig. 7 presents an example of the comparison of the signature portion signal obtained from different placements. Even though the same user writes the same signature, the signature portion obtained from different placement is quite different. It is hard to distinguish the signature portion from the pulse signal in the case of *radial artery* or *posterior wrist-side*. On the contrary, with the placement at *median antebrachial vein*, the signature portion is distinguishable among the pulse signals. Such placement is also feasible since the user can rotate the smartwatch/tracker $180^o$ so that its PPG sensor faces the anterior side of the wrist when writing the signature.

**Attack from Same User:** This is a unique type of attack where the user himself is trying to falsify signing the document in the system by signing other than his signature. For example,



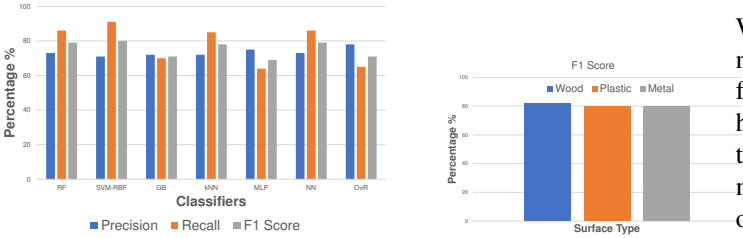

(a) segmentation algorithms (b) classifiers (c) training size

Fig. 6. Performance of PPGSign for (a) different segmentation algorithms (average over all classifiers with 60% training data), (b) different classifiers (Skewness-DTW segmentation with 80% training data), (c) training size is varying from 20%-80% (average over all classifiers).




(a) median antebrachial vein (b) radial artery (c) posterior wrist-side

Fig. 7. Signature portion of PPG signals collected by PPG sensor at different locations: (a) median antebrachial vein, (b) radial artery, (c) posterior wrist-side.




(a) the same user attack (b) different surfaces

Fig. 8. Performance of PPGSign (a) under attacks from the same user, (b) on three different surfaces - wood, metal, and plastic.

a user is legitimately ordered to sign a document which he/she does not want to sign. So the user tries to forge his own signature to escape from the situation. Our system shows a promising performance to handle this type of attack. Here, for each user's dataset, only his invalid signatures are used as the forged signatures. The result is shown in Fig. 8(a), with near 80% scores for 20% training data.

**Impact of Surface:** The matter of surface comes into place when the user is signing on a document placed on a certain surface. Though the signing happens on a piece of paper, the surface the paper is placed on might be different such as, wood, plastic, and metal. We have collected 30 new samples for each type of surface. Fig. 8(b) shows the results for different surfaces. Obviously, PPGSign's performance is almost the same in each of the scenarios. There is no impact of surfaces on the system unless the surface is really uneven, for which the finger and wrist movements are irregular.

## V. Conclusion

In this paper, we have proposed PPGSign, a non-intrusive and secure handwritten signature authentication system, which leverages PPG sensors in the existing wearable devices. The system exploits the unique blood flow volume changes in the users' hand movement to validate the handwritten signature. We built a low-cost hardware prototype using COTS components and evaluate its performance via experiments. The results from our evaluation confirm the feasibility of the PPG-based handwritten signature authentication. In the future, we will test our system with more data/users, further investigate other machine learning techniques, and explore real implementation on the commercial smart band and develop a user-friendly mobile app for real-world applications.

## References

[1] D. Impedovo and G. Pirlo, "Automatic signature verification: The state of the art," *IEEE TSMC, Part C*, vol. 38, no. 5, pp. 609–635, 2008.
[2] M. A. Ferrer, J. F. Vargas, A. Morales, and A. Ordonez, "Robustness of offline signature verification based on gray level features," *IEEE Trans. on Information Forensics and Security*, vol. 7, no. 3, pp. 966–977, 2012.
[3] K. Yasuda, et al., "Visual-based online signature verification using features extracted from video," *J. Net & Comp App*, vol.33, p.333, 2010.
[4] N. Paudel, M. Querini, and G. F. Italiano, "Handwritten signature verification for mobile phones." in *Proc. of ICISSP*, 2016.
[5] A. Levy, B. Nassi, et al., "Handwritten signature verification using wrist-worn devices," *Proc. ACM IMWUT*, vol. 2, no. 3, 2018.
[6] G. Li and H. Sato, "Handwritten signature authentication using smart-watch motion sensors," in *Proc. of IEEE COMPSAC*, 2020.
[7] M. Chen, J. Lin, et al., "Silentsign: Device-free handwritten signature verification through acoustic sensing," in *Proc. of IEEE PerCom*, 2020.
[8] Y. Cao, H. Chen, F. Li, and Y, Wang, "Crisp-BP: Continuous wrist PPG-based blood pressure measurement," in *ACM MobiCom* 2021.
[9] Y. Cao, et al., "PPGPass: Nonintrusive and secure mobile two-factor authentication via wearables," in *Proc. of IEEE INFOCOM*, 2020.
[10] T. Zhao, et al., "TrueHeart: Continuous authentication on wrist-worn wearables using PPG-based biometrics," in *IEEE INFOCOM*, 2020.
[11] A. Savitzky, M.J.E. Golay, "Smoothing and differentiation of data by simplified least squares procedures." *Analy. Chem.*, vol.36, p.1627, 1964.
[12] T. Zhao, J. Liu, et al., "PPG-based finger-level gesture recognition leveraging wearables," in *Proc. of IEEE INFOCOM*, 2018.
[13] C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," *Signal Processing*, vol. 167, p. 107299, 2020.
[14] Y. Benjamini and D. Yekutieli, "The control of the false discovery rate in multiple testing under dependency," *Ann. Stat.*, vol. 29, 2001.
[15] M. Christ, N. Braun, J. Neuffer, and A. Kempa-Liehr, "Time series feature extraction on basis of scalable hypothesis tests (tsfresh – a python package)," *Neurocomputing*, 05, 2018.
[16] F. Pedregosa, et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.