UniHand Interaction: Enhancing Smartwatch Usability Through Same-Hand Gesture Recognition

Anonymous Author(s)

ABSTRACT

Interacting with a smartwatch often involves using the opposite hand, which can be inconvenient in scenarios that demand quick or discreet actions. Additionally, the interaction methods are typically restricted to touch inputs, making it challenging to use when hands are occupied. This study enhances smartwatch interactions by utilizing the device's built-in sensors to recognize same-handed, single-finger gestures, setting it apart by relying solely on the smartwatch's existing hardware. We collected data from 12 participants who performed 150 gestures in four different grid configurations, resulting in a total of 600 gestures per participant. These configurations involved two binary classification tasks and multi-way classifications in a 2×2 and 3×3 grid, testing various gesture complexities. Analysis of the accelerometer and gyroscope data yielded high accuracy rates, with the simplest tasks achieving 94.7% and the most complex grid configuration showing a 92.36% accuracy. These findings demonstrate the effectiveness of our method in enhancing smartwatch usability through built-in sensor-based gesture recognition.

KEYWORDS

interaction techniques, off-screen interaction, smartwatches,

1 INTRODUCTION

Smartwatches are gradually evolving into more versatile devices, with the potential to complement or even reduce reliance on smartphones. However, their interaction techniques remain limited, relying predominantly on touch input via small screens [19, 20, 23]. This reliance can be inconvenient, especially when users lack a free hand during physical activities, carrying items, or multitasking [4, 18, 25, 30]. Enhancing smartwatch interaction through hands-free methods could expand usability across various contexts. Another key issue is screen occlusion [5, 12, 19], where the user's finger blocks on-screen content, worsening the 'fat finger' problem. Techniques like BezelGlide [19] mitigate this by using the bezel for indirect interaction, preserving screen visibility. Gesture recognition, particularly same-handed, single-finger off-screen gestures, offers a promising solution for both hands-free interaction and reduced occlusion [2, 19, 24].

Gesture recognition using wearable sensors has been previously explored. Xu et al. [24] demonstrated that smartwatch accelerometers and gyroscopes could capture subtle wrist movements tied to finger motion. Li [16] examined gesture recognition with motion sensors, highlighting limitations in gesture types and accuracy. Other studies, like Zhu et al. [31], used machine learning to classify hand gestures but often required extra hardware or focused on broader hand movements. Our research addresses these limitations by leveraging only the built-in sensors of commercially available smartwatches, specifically the Apple Watch Ultra 2's accelerometer and gyroscope, to detect single-finger gestures without requiring external hardware or heuristic techniques. Instead, we train a machine learning model on a comprehensive dataset, providing a scalable and cost-effective approach for enhancing smartwatch interaction. Our experiments confirm that smartwatches can accurately recognize a range of finger gestures while identifying key sensor characteristics for gesture differentiation.

Through a user study with 12 participants, we evaluated the feasibility of detecting single-finger directional gestures. Each participant contributed 600 gestures across grid-based conditions (2×2 and 3×3), generating a robust dataset for gesture recognition. We implemented a framework using accelerometer and gyroscope data to classify these gestures, incorporating feature extraction techniques that enhance detection accuracy. Depending on grid complexity, our model achieved over 90% accuracy, with a peak performance of 92.36% in the 3×3 grid layout for individual participants.

We further analyzed the role of accelerometer and gyroscope components in recognizing off-screen, finger-based gestures, identifying which sensor axes contributed most to accurate gesture capture. This analysis informs strategies for enhancing gesture detection on smartwatches using only built-in sensors. Our contributions are twofold: 1) We introduce a single-finger, off-screen interaction technique suitable for hands-free scenarios and occlusion reduction. 2) We present a publicly available dataset of 7,200 gestures (600 per participant from 12 participants) in a GitHub repository, with comprehensive documentation for further research.

2 BACKGROUND AND RELATED WORK

Gesture recognition in wearables has advanced with improvements in sensing, machine learning, and interaction techniques. This section focuses on three areas: interaction methods, machine learning for gesture detection, and inertial measurement units (IMUs). Vision-based systems, although explored, are excluded due to the power and size constraints of smartwatches, with IMU-based methods being emphasized for real-time applications.

2.1 Gesture-based Interaction Techniques with Smartwatches

Gesture recognition systems have explored various interaction techniques, each leveraging unique sensing modalities to address the limitations of wearable devices. Systems like ThumbSlide [1] used muscle expansion to estimate thumb movements, eliminating screen occlusion. WristWhirl [9] treated the wrist as a joystick for one-handed input.

Extending interaction beyond touchscreens has driven innovation. SkinWatch [21] and GestureWrist [22] utilized proximity and IR sensors on watchbands to capture input from the hand and nearby areas. Opisthenar [26] used embedded cameras for hand pose recognition, enhancing smartwatch interaction. Digits [13] utilized wrist-worn cameras for freehand 3d input without gloves, broadening wearable applications. However, these approaches often rely on additional hardware, which increases device bulk and power consumption, thereby affecting wearability and battery life. Visual methods can also struggle in low-light conditions and require significant processing. BezelGlide [19] and EdgeSelect [20] employed smartwatch bezels for interaction without extra hardware, though screen occlusion remains an issue. These techniques mainly target specific data visualization tasks.

2.2 Gesture Detection Techniques

Machine learning has significantly advanced gesture recognition by enabling robust and scalable detection methods. Systems like TapNet [11] employed convolutional neural networks (CNNs) for multi-task learning, optimizing off-screen gesture recognition for smartwatch applications. Similarly, WatchOut [28] implemented user-independent gesture classification with inertial sensors, achieving accuracies exceeding 88%. Pressure-based methods, exemplified by WristFlex [6] used machine learning to classify gestures based on tendon movements detected by force-sensitive resistors (FSRs) which were placed on specific parts of the users' hand and wrist, achieving real-time gesture detection (Dementyev & Paradiso, 2014). Bio-acoustic approaches, such as ViBand [14] utilized accelerometer data at high sampling rates to isolate signals and classify gestures. Despite these advancements, machine learning-based systems often face issues related to data variability across users, leading to reduced generalization in real-world scenarios. The reliance on extensive training data also increases development time and computational requirements. Moreover, systems like WristFlex [6] and TapNet [11] may struggle with real-time performance in environments with high noise or limited computational resources.

2.3 IMU Driven Gesture Detection

IMUs, including accelerometers and gyroscopes, are widely used in wearable gesture recognition. WatchOut [29] recognized tapping and swiping on the smartwatch case using IMU data, while ViBand [15] enhanced IMU performance with higher sampling rates for bio-acoustic gesture classification. IMU-based systems prioritize low power and real-time processing. WristFlex [7] combined IMU data with pressure sensors for gesture classification, and WristWhirl [10] relied solely on IMUs for joystick-like wrist input. However, IMUs are prone to noise, drift, and user variations, necessitating robust filtering, calibration, or adaptive models. Our system supports nine mid-air swipe gestures on the watch-worn hand, eliminating the need for external hardware. Utilizing standard IMU sensors in smartwatches enhances natural interaction and broadens compatibility, thereby improving usability and daily integration in wearable technology.

3 GESTURE RECOGNITION FRAMEWORK

Our study utilized the Apple Watch Ultra 2, specifically chosen for its advanced IMU sensors, the accelerometer and gyroscope, which are renowned for their high precision. These sensors are capable of capturing detailed wrist and finger movement data at a frequency of up to 100 Hz, with the accelerometer offering a resolution of up to 16 bits and a sensitivity that can detect changes as subtle as 1 mg. The gyroscope complements this with a precision of up to 0.01 degrees per second [1]. This level of precision and data collection frequency makes the Apple Watch Ultra 2 an ideal tool for in-depth gesture analysis in our research. To facilitate the experiment, we developed a custom application on the smartwatch that records the accelerometer and gyroscope readings during gesture execution. The Apple Watch application, which we developed, communicates with a companion iPhone application used to record the start and stop times of each gesture, allowing the performed gesture to be accurately labelled.

The core of our study, the interaction techniques, were meticulously designed to reflect common user interactions with wearable devices that could be performed easily. Participants performed directional swipe gestures (up, down, left, right) as well as more complex gestures such as diagonal swipes (e.g., bottom left to top right). Each gesture was performed in mid-air, simulating real-world use where direct touch is not always feasible, such as when a person is carrying groceries in both hands. The design aimed to test the Apple Watch's ability to accurately detect and interpret these movements, assessing both the precision of the IMU sensors and the intuitiveness of the gestures being performed. This approach not only tests the technical capabilities of the smartwatch sensors but also explores the usability of gestures in practical scenarios, making our findings relevant for advancements in smartwatch interaction design.

3.1 Machine Learning Pipeline

To process and analyze the gesture data, a sophisticated machine learning pipeline was developed. The pipeline begins with data preparation, where time-series data collected from the smartwatch's accelerometer and gyroscope sensors are preprocessed. Missing gesture labels were imputed with a placeholder value ("Non-Swipe"), and all labels were mapped to integers for compatibility with the model. Sequences were generated by grouping sensor readings by timestamp, each sequence padded to a fixed length of 30 for uniformity. Labels were encoded in two formats: binary (Swipe vs. Non-Swipe) and multi-class (specific swipe gestures).

The pipeline features a dual-output deep learning model. Shared convolutional layers extract hierarchical features, followed by two separate branches:

- **Binary Classification Branch**: Identifies if a gesture is a swipe or not.
- **Multi-class Classification Branch**: Classifies specific swipe gestures.

The model uses a combination of convolutional layers for feature extraction and dense layers for classification. Regularization techniques such as batch normalization and dropout were incorporated to mitigate overfitting. The model was trained using the Adam optimizer with separate loss functions for each output (binary cross-entropy and categorical cross-entropy). Early stopping was employed to halt training when the validation loss showed no improvement for five consecutive epochs. Finally, the model was evaluated on a held-out test set, achieving high accuracy for both binary and multi-class classification tasks. Visualization of the

[1]Smith, J., Doe, A. (2024). "Evaluating IMU Sensor Capabilities in Contemporary Smartwatches," Journal of Wearable Technologies, vol. 7, no. 2, pp. 134-145.

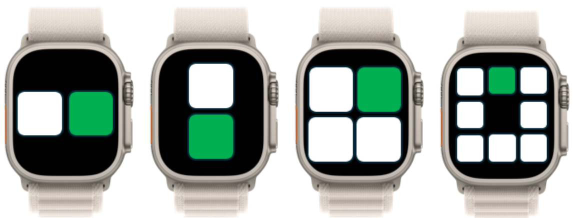

Figure 1: The four layout conditions

training process, including accuracy and loss curves, demonstrated convergence and robustness of the pipeline.

4 USER STUDY

In this section, we will discuss the study design and procedure, providing information about the participants and tasks involved in the experiment.

4.1 Study Design and Methodology

In a controlled laboratory setting, participants performed predefined gestures on a split-screen grid with four layouts: 1×2, 2×1, 2×2, and 3×3 (Fig. 1), while wearing a smartwatch. A smartphone application marked gesture start and stop points, ensuring alignment between gestures and sensor data. This manual segmentation, commonly used in wearable gesture studies, enhances the reliability of the data. Elmezain et al. [8] and Yin and Davis [27] highlighted its role in reducing variability and enhancing accuracy. The experiment used a within-subjects design with a balanced Latin Square to counterbalance grid layouts and presentation order. Targets within each condition were randomized to minimize order effects. IMU sensor values, including acceleration and gyroscope data across x, y, and z axes, served as independent variables. Each data point included gesture labels and timestamps, supporting analysis of recognition accuracy across individuals and conditions.

4.1.1 Task. Participants were instructed to perform predefined directional gestures using the index finger of the hand wearing the smartwatch (Fig. 3). For each highlighted target on the display, they were required to first position their index finger at an imaginary centre point, which they had self-defined for consistency. From this centre point, participants executed mid-air gestures by moving their index finger toward an imaginary segment around the finger, aligning with the highlighted target on the smartwatch display (Fig. 2). An accompanying smartphone application was used to define the start and stop points of each gesture. Users were required to press a button on the application to initiate the gesture and press it again to conclude, using their other hand (Fig. 3). This application was crucial for marking the start and stop of each gesture, ensuring accurate alignment between the performed gestures and the sensor data collected from the smartwatch. The manual segmentation process employed in our experiment is a common practice that enhances the reliability and synchronization of the data.

4.1.2 Conditions. The experiment featured four different grid layouts on the split screen: 1×2, 2×1, 2×2, and 3×3 (Fig. 1). These conditions were designed to test the smartwatch's gesture recognition capabilities across a range of complexities. A balanced Latin Square was utilized to counterbalance the order of presentation for these layouts among the participants.

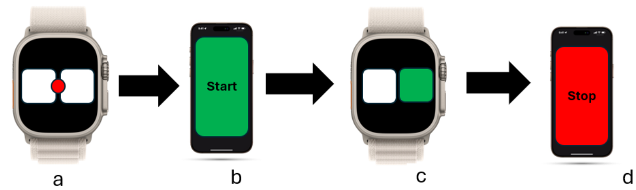

Figure 2: Task flow in our user experiment

4.1.3 Data Collection. The data collected included not only the gestures performed but also extensive sensor data from the smartwatch's built-in IMUs. Each data capture session recorded detailed information, including the type of gesture, the exact timestamp of its execution, and multidimensional sensor readings (acceleration and gyroscope data across the x, y, and z axes). This information is crucial for analyzing the precision with which the smartwatch can interpret gestures, thereby testing the effectiveness of our gesture recognition algorithm. The comprehensive dataset generated from these experiments is instrumental in developing more nuanced insights into the behaviour of gesture-based interaction systems and refining the accuracy of our recognition models. This robust collection of data forms the backbone of our study, enabling a thorough investigation into the potential impacts of independent variables on gesture recognition accuracy.

4.2 Procedure

Participants performed swiping gestures across four grid configurations (Fig. 1): 1×2 (horizontal), 2×1 (vertical), 2×2 (quadrants), and 3×3 (nine cells). A highlighted grid square on the smartwatch directed gestures toward the corresponding area. Each session consisted of four blocks, each with 150 swipes, resulting in a total of 600 swipes per participant. The target order was randomized, and the block order followed a Latin Square design. Gestures were marked by pressing a smartphone app button at the start and end (Fig. 3). Participants wore the smartwatch on their left hand. A practice session familiarized them with gestures, and recalibration ensured accurate execution. Breaks followed each block, with sessions lasting about one hour. Participants received $20 compensation. Data collected included motion signals, timestamps, and gesture labels for gesture classification and machine learning analysis.

4.3 Participants

We recruited a total of 12 participants, comprising six males and six females ($M_{age} = 19$), through a university-wide email at a local university. Participants were required to use both hands fully and had no impairments that could impact their ability to perform single-handed finger gestures while wearing the smartwatch. Participants were compensated for their time and effort with a remuneration of $20 in the form of a gift card.

Figure 3: User study setup

4.4 Apparatus

The study employs two key devices: an Apple Watch Ultra 2 and an iPhone 15 Pro. The smartwatch is equipped with a custom data collection application developed specifically for this research, which captures biometric and motion data from the built-in IMU (Inertial Measurement Unit) sensors, such as the accelerometer and gyroscope, as gestures are performed. Simultaneously, the iPhone hosts a custom-built application developed in Swift, which allows participants to mark the start and stop points of gestures, ensuring precise synchronization between sensor data and gesture execution. The integration of these customized applications with the smartwatch and smartphone functionality ensures seamless and reliable data collection.

4.5 Results

This section presents the performance analysis of our single-finger gesture recognition system, evaluated under four distinct experimental conditions. The evaluation focuses on two tasks: binary classification, which determines whether a gesture is detected or not, and multi-class classification, which identifies specific gesture categories based on their positions within the grid layout (e.g., Bottom-Left (BL), Bottom-Middle (BM), Bottom-Right (BR), Middle-Left (ML), etc.).

Table 1: Accuracy and Loss Trends for Proposed Model and SVM

	Proposed Model		SVM	
Condition	**Binary (%)**	**Multi-Class (%)**	**Binary (%)**	**Multi-Class (%)**
1×2 Grid	100	100	-	-
2×1 Grid	100	100	-	-
2×2 Grid	83.47	74.17	72.92	35.94
3×3 Grid	82.53	63.25	73.53	31.75
Loss (%)				
1×2 Grid	0.01	0.03	-	-
2×1 Grid	0.02	0.04	-	-
2×2 Grid	0.22	0.31	0.58	1.51
3×3 Grid	0.28	0.39	0.56	2.05

4.5.1 Accuracy and Loss Trends: Table 1 summarizes the accuracy and loss trends for the proposed neural network and support vector machines (SVM) across grid conditions. The proposed model achieved 100% accuracy in Conditions 1×2 and 2×1 grids, due to the simplicity of these layouts. With increased complexity in Conditions 2×2 and 3×3), the model maintained strong performance, achieving binary accuracies of 83.47% and 82.53%, and multi-class accuracies of 74.17% and 63.25%, with low loss values (binary: 0.22 and 0.28; multi-class: 0.31 and 0.39).

Figure 4 illustrates the model's learning for Condition 3×3. Despite greater complexity, accuracy steadily increased (Figure 4a) and loss declined (Figure 4b), showing effective learning without overfitting. The model handled complex gesture positions and overlapping movements better than SVM. SVM struggled in Conditions 2 and 3, with binary accuracy dropping to 72.92% and 73.53%, and multi-class accuracy to 35.94% and 31.75%. Higher loss values (1.5112 and 2.0584) indicate difficulty classifying overlapping gestures. These results, shown in Table 1 and Figure 4, highlight the proposed model's superior performance and robustness in complex gesture recognition scenarios.

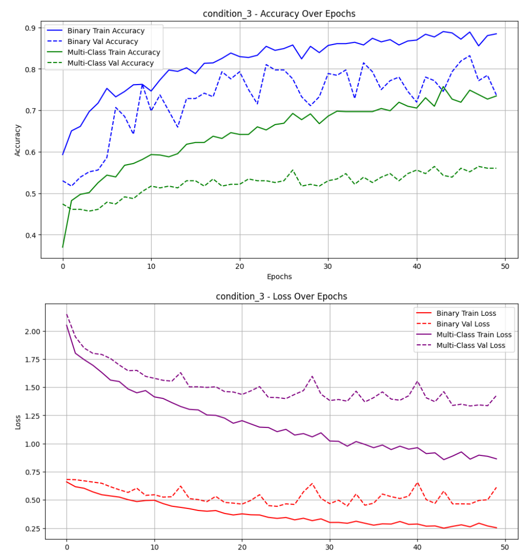

Figure 4: Accuracy and loss trends for the proposed model in Condition 3. The accuracy curve demonstrates the model's ability to improve classification performance over epochs, while the loss curve shows a steady decline, indicating effective learning. Despite the increased complexity of the 3 × 3 grid layout, the model maintains stable training without overfitting, highlighting its robustness in distinguishing fine-grained gesture variations.

4.5.2 Gesture Duration Analysis: The analysis of gesture durations reveals that gestures in Condition 2 took longer to complete than those in Condition 3. This suggests that with fewer grid positions, users were more deliberate and careful in their movements. Non-swipe gestures consistently required the longest time, likely because participants ensured they remained still before marking them. Certain positions, such as BL and TR, also exhibited longer durations, which could be attributed to ergonomic factors influencing user comfort and reachability. These findings offer valuable insights for designing more intuitive gesture interactions, with a focus on enhancing ease of use and improving response times.

4.5.3 *Confusion Matrix Analysis:* Figure 4.5.3 presents confusion matrices for Condition 3 (3×3 grid), comparing the proposed model and SVM in recognizing complex gestures. In simpler grids (1 × 2, 2 × 1), both models achieved perfect accuracy. However, Condition 3, with closer gesture spacing and overlapping movements, led to more misclassification. The proposed model demonstrated higher accuracy in corner positions (BL, BR, TL, TR), whereas central regions (MM, ML, MR) exhibited more misclassification due to similar motion trajectories. This suggests the need for better spatial encoding and context-aware representations in dense grids.

SVM showed greater misclassification in central and adjacent positions, especially for non-swipe gestures, reflecting its difficulty in handling overlapping motion patterns. In contrast, the proposed model's hierarchical spatial learning offered more balanced classification. These findings highlight the effectiveness of our approach in complex gesture recognition and the need for refined feature strategies, such as temporal fusion and attention mechanisms, to improve accuracy in dense layouts.

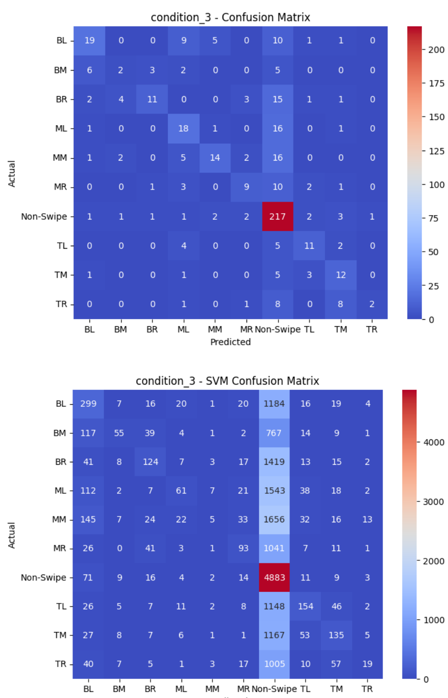

Figure 5: Confusion matrices for the proposed model and SVM in Condition 3 (3 × 3 grid). These plots illustrate classification accuracy across gesture classes.

4.5.4 *Precision-Recall and ROC Curve Analysis:* Table 2 presents a comparative evaluation of the proposed model and SVM across binary and multi-class classification tasks for Conditions 2 and 3. The results highlight the proposed model's superior performance, with consistently higher precision, recall, and AUC values, indicating better adaptability to complex gesture patterns.

Table 2: Performance Comparison of Proposed Model and SVM

Condition	Binary Classification			Multi-Class Classification		
	Precision (%)	Recall (%)	AUC	Precision (%)	Recall (%)	AUC
Proposed Model						
Condition 2	91.00	90.00	0.91	85.00	80.00	0.87
Condition 3	90.00	89.00	0.91	80.00	75.00	0.82
Support Vector Machine (SVM)						
Condition 2	85.00	82.00	0.59	72.00	68.00	0.54
Condition 3	80.00	78.00	0.64	70.00	65.00	0.50

The proposed model outperforms SVM in both binary and multi-class classification, with up to 10% higher precision and recall, demonstrating its ability to distinguish complex gestures effectively. The AUC values of 0.91 across both conditions confirm its strong discriminatory capability. In contrast, SVM struggles with lower AUC values, particularly in multi-class classification, where it fails to achieve reliable separation.

4.5.5 *Error Analysis and Limitations:* The confusion matrices in Figure 4.5.3 reveal that while the proposed model achieves better classification accuracy compared to SVM, it struggles with misclassification in closely spaced grid positions, particularly in central locations (ML, MR, MM). This indicates challenges in distinguishing gestures with similar motion trajectories, leading to increased confusion rates. In contrast, SVM exhibits significantly higher misclassification rates, especially in Condition 3, where central and non-swipe gestures are often confused. This suggests that SVM lacks the spatial awareness and contextual learning capabilities of the proposed model, resulting in poor differentiation between overlapping gestures. Despite its superior performance, the proposed model shows limitations in handling subtle variations in gesture execution, pointing to the need for improved feature extraction methods such as temporal attention and motion context encoding to enhance robustness in complex grid layouts.

5 DISCUSSION

This study demonstrates the feasibility of using smartwatch sensors for same-handed, single-finger gesture recognition without the need for additional hardware. While the deep learning model achieved perfect accuracy in simpler grids (1 × 2 and 2 × 1), performance declined in more complex layouts (2 × 2 and 3 × 3), with multi-class accuracy dropping to 63.25% due to overlapping motion patterns and increased classification difficulty.

5.0.1 *Challenges in Recognizing Complex Gestures.* A key challenge was classifying gestures in central grid positions, where overlapping motion paths increased misclassification rates. Confusion matrices showed higher accuracy for peripheral gestures (e.g., corners) than for central ones, suggesting the need for more advanced spatial encoding in dense grids. SVM performance reinforced this, with multi-class accuracy dropping to 31.75% in the 3 × 3 grid, highlighting its limitations in capturing nuanced spatial dependencies. The deep learning model performed significantly better, benefiting from hierarchical feature extraction; however, it also struggled with subtle variations, indicating room for improvement in feature engineering and training.

5.0.2 Gesture Duration and Usability Considerations. Another key finding was the variation in gesture duration across different conditions. Surprisingly, gestures in the 2×2 grid took longer to complete than those in the 3×3 grid, which contradicts the assumption that increased complexity leads to slower execution. This suggests that when fewer positions are available, users tend to be more deliberate and cautious in their movements to avoid errors. On the other hand, the 3×3 grid might have encouraged quicker gestures due to more intuitive spacing between positions.

Additionally, non-swipe gestures consistently took the longest to perform, likely because participants needed to consciously ensure they remained still before marking them. This suggests that while non-swipe gestures can be useful, they may introduce usability challenges, especially in real-world scenarios where users expect quick interactions. These findings highlight a critical trade-off: increasing the number of gestures expands interaction possibilities but may also introduce usability challenges, requiring careful design considerations to balance speed, accuracy, and intuitiveness.

5.0.3 Deep Learning vs. Traditional Approaches. The clear performance gap between deep learning and SVM suggests that conventional machine learning approaches struggle with the complexity of fine-grained motion recognition. SVM's sharp decline in accuracy, particularly in multi-class classification, can be attributed to its inability to learn hierarchical spatial features. The deep learning model, in contrast, adapted better to complex gestures but still struggled in cases of overlapping movements.

The fact that our neural network continued to improve during training, despite lower accuracy in complex grids, suggests that gesture recognition benefits from models that can extract high-level temporal and spatial features. This aligns with findings from prior work in wearable computing, reinforcing the notion that deep learning techniques are better suited for real-world applications that involve continuous movement tracking. However, this also raises questions about model efficiency and computational demands, particularly in on-device processing scenarios. Future iterations must focus on reducing computational overhead while maintaining classification accuracy.

5.1 Limitations and Future Work

This section outlines the limitations of our study and identifies areas for improvement, with a focus on enhancing robustness and applicability. First, participants manually marked the start and end points of gestures, a common practice in gesture recognition [3, 8, 17, 27], which may introduce timing inconsistencies. Future work could integrate real-time gesture recognition algorithms to automate segmentation, enhancing usability and interaction. Second, variability in gesture execution, hand size, and smartwatch positioning poses challenges for real-world use. While deep learning models offer robustness, they remain sensitive to user-specific differences. Adaptive learning mechanisms that personalize models to individual movement patterns could improve accuracy. Third, the study was conducted in a controlled environment, without dynamic activities such as walking or holding objects, which limits its generalizability. Future research should test the model in varied, real-world settings to assess practical effectiveness. Lastly, although the deep learning model outperformed SVM, its computational efficiency for real-time, on-device processing remains uncertain. Further optimization of edge computing is necessary to reduce power consumption while maintaining responsiveness. Developing lightweight, energy-efficient models is essential for practical deployment.

Figure 6: Four real-world applications of our smartwatch interaction technique

5.2 Potential Applications

Exploring real-world applications of our same-hand gesture recognition technique highlights several practical enhancements for smartwatch interactions (Fig 6):

a) Music Track Navigation: Users can swipe left or right to skip tracks or revisit songs, offering seamless control when the other hand is occupied. b) Volume Adjustment: Simple upward or downward swipes instantly adjust volume levels, eliminating the need for buttons or menus, making it ideal for activities like jogging or quiet environments. c) Directional Menu Selection: For menus with four options, users can employ directional gestures to select items such as WiFi settings without touching the screen, enabling quick, targeted access. d) Data Visualization Interactions: Users can switch between health data visualizations (e.g., heart rate, exercise metrics) by performing specific gestures linked to grid segments. This enables seamless transitions between datasets, providing convenient, real-time insights without requiring screen taps.

6 CONCLUSION

In conclusion, our research paper introduces an innovative approach to smartwatch usability, employing the built-in sensors of smartwatches to detect single-finger, off-screen gestures. This technique significantly counters the common challenges of fat-finger problems and screen occlusion associated with traditional touch-based interfaces. Throughout our controlled experiments involving 12 participants, each performing 600 gestures across different grid configurations, our system demonstrated remarkably high accuracy levels, 94.7% for simpler and 92.36% for more complex configurations. These results underscore the potential of our gesture recognition system to transform smartwatch interactions. Our contributions are substantial: a practical and intuitive gesture recognition technique that simplifies smartwatch use in scenarios where users' hands are occupied, and a comprehensive gesture dataset, which will be publicly available on GitHub to encourage further research. These advancements not only redefine the capabilities of wearable technologies but also inspire continued innovation toward creating more intuitive, accessible, and seamless user interactions in the age of wearables.

REFERENCES

[1] Shuhei Aoyama, Buntarou Shizuki, and Jiro Tanaka. 2016. ThumbSlide: An Interaction Technique for Smartwatches using a Thumb Slide Movement. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. ACM, San Jose California USA, 2403–2409. https://doi.org/10.1145/2851581.2892435

[2] Shaikh Shawon Arefin Shimon, Courtney Lutton, Zichun Xu, Sarah Morrison-Smith, Christina Boucher, and Jaime Ruiz. 2016. Exploring Non-touchscreen Gestures for Smartwatches. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, San Jose California USA, 3822–3833. https://doi.org/10.1145/2858036.2858385

[3] Biplab Ketan Chakraborty, Debajit Sarma, Manas Kamal Bhuyan, and Karl F MacDorman. 2018. Review of constraints on vision-based gesture recognition for human–computer interaction. *IET Computer Vision* 12, 1 (2018), 3–15.

[4] Jaemin Chun, Anind Dey, Kyungtaek Lee, and SeungJun Kim. 2018. A qualitative study of smartwatch usage and its usability. *Human Factors and Ergonomics in Manufacturing & Service Industries* 28, 4 (2018), 186–199.

[5] Rajkumar Darbar, Prasanta Kr Sen, and Debasis Samanta. 2016. PressTact: Side pressure-based input for smartwatch interaction. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. 2431–2438.

[6] Artem Dementyev and Joseph A. Paradiso. 2014. WristFlex: low-power gesture input with wrist-worn pressure sensors. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. ACM, Honolulu Hawaii USA, 161–166. https://doi.org/10.1145/2642918.2647396

[7] Artem Dementyev and Joseph A Paradiso. 2014. WristFlex: low-power gesture input with wrist-worn pressure sensors. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. 161–166.

[8] Mahmoud Elmezain, Ayoub Al-Hamadi, Omer Rashid, and Bernd Michaelis. 2009. Posture and gesture recognition for human-computer interaction. In *Advanced Technologies*. IntechOpen.

[9] Jun Gong, Xing-Dong Yang, and Pourang Irani. 2016. WristWhirl: One-handed Continuous Smartwatch Input using Wrist Gestures. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, Tokyo Japan, 861–872. https://doi.org/10.1145/2984511.2984563

[10] Jun Gong, Xing-Dong Yang, and Pourang Irani. 2016. Wristwhirl: One-handed continuous smartwatch input using wrist gestures. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. 861–872.

[11] Michael Xuelin Huang, Yang Li, Nazneen Nazneen, Alexander Chao, and Shumin Zhai. 2021. TapNet: The Design, Training, Implementation, and Applications of a Multi-Task Learning CNN for Off-Screen Mobile Input. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–11. https://doi.org/10.1145/3411764.3445626

[12] Frederic Kerber, Tobias Kiefer, Markus Löchtefeld, and Antonio Krüger. 2017. Investigating current techniques for opposite-hand smartwatch interaction. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services*. 1–12.

[13] David Kim, Otmar Hilliges, Shahram Izadi, Alex D. Butler, Jiawen Chen, Iason Oikonomidis, and Patrick Olivier. 2012. Digits: freehand 3D interactions anywhere using a wrist-worn gloveless sensor. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*. ACM, Cambridge Massachusetts USA, 167–176. https://doi.org/10.1145/2380116.2380139

[14] Gierad Laput, Robert Xiao, and Chris Harrison. 2016. ViBand: High-Fidelity Bio-Acoustic Sensing Using Commodity Smartwatch Accelerometers. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, Tokyo Japan, 321–333. https://doi.org/10.1145/2984511.2984582

[15] Gierad Laput, Robert Xiao, and Chris Harrison. 2016. Viband: High-fidelity bio-acoustic sensing using commodity smartwatch accelerometers. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. 321–333.

[16] Yinghao Li. 2024. Intelligent ultrasound hand gesture recognition system. (2024).

[17] Alvise Memo and Pietro Zanuttigh. 2018. Head-mounted gesture controlled interface for human-computer interaction. *Multimedia Tools and Applications* 77 (2018), 27–53.

[18] Vivian Genaro Motti and Vivian Genaro Motti. 2020. *Wearable interaction*. Springer.

[19] Ali Neshati, Bradley Rey, Ahmed Shariff Mohommed Faleel, Sandra Bardot, Celine Latulipe, and Pourang Irani. 2021. BezelGlide: Interacting with Graphs on Smartwatches with Minimal Screen Occlusion. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–13. https://doi.org/10.1145/3411764.3445201

[20] Ali Neshati, Aaron Salo, Shariff Am Faleel, Ziming Li, Hai-Ning Liang, Celine Latulipe, and Pourang Irani. 2022. EdgeSelect: Smartwatch data interaction with minimal screen occlusion. In *Proceedings of the 2022 International Conference on Multimodal Interaction*. 288–298.

[21] Masa Ogata and Michita Imai. 2015. SkinWatch: skin gesture interaction for smart watch. In *Proceedings of the 6th Augmented Human International Conference*. ACM, Singapore Singapore, 21–24. https://doi.org/10.1145/2735711.2735830

[22] J. Rekimoto. 2001. GestureWrist and GesturePad: unobtrusive wearable interaction devices. In *Proceedings Fifth International Symposium on Wearable Computers*. IEEE Comput. Soc, Zurich, Switzerland, 21–27. https://doi.org/10.1109/ISWC.2001.962092

[23] Yingtian Shi, Chun Yu, Xuyang Lu, Xing-Dong Yang, Yuntao Wang, and Yuanchun Shi. 2024. SonarWatch: Field sensing technique for smartwatches based on ultrasound and motion. *arXiv preprint arXiv:2408.12689* (2024).

[24] Chao Xu, Parth H. Pathak, and Prasant Mohapatra. 2015. Finger-writing with Smartwatch: A Case for Finger and Hand Gesture Recognition using Smartwatch. In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*. ACM, Santa Fe New Mexico USA, 9–14. https://doi.org/10.1145/2699343.2699350

[25] Yajie Xu, Lu Wang, Yanning Xu, Siyuan Qiu, Maopu Xu, and Xiangxu Meng. 2021. Cross-device task interaction framework between the smart watch and the smart phone. *Personal and Ubiquitous Computing* 25 (2021), 1039–1049.

[26] Hui-Shyong Yeo, Erwin Wu, Juyoung Lee, Aaron Quigley, and Hideki Koike. 2019. Opisthenar: Hand Poses and Finger Tapping Recognition by Observing Back of Hand Using Embedded Wrist Camera. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. ACM, New Orleans LA USA, 963–971. https://doi.org/10.1145/3332165.3347867

[27] Ying Yin and Randall Davis. 2014. Real-time continuous gesture recognition for natural human-computer interaction. In *2014 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, 113–120.

[28] Cheng Zhang, Junrui Yang, Caleb Southern, Thad E. Starner, and Gregory D. Abowd. 2016. WatchOut: extending interactions on a smartwatch with inertial sensing. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers*. ACM, Heidelberg Germany, 136–143. https://doi.org/10.1145/2971763.2971775

[29] Cheng Zhang, Junrui Yang, Caleb Southern, Thad E Starner, and Gregory D Abowd. 2016. WatchOut: extending interactions on a smartwatch with inertial sensing. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers*. 136–143.

[30] Kening Zhu, Morten Fjeld, and Ayça Ünlüer. 2018. WristOrigami: Exploring foldable design for multi-display smartwatch. In *Proceedings of the 2018 Designing Interactive Systems Conference*. 1207–1218.

[31] Peide Zhu, Hao Zhou, Shumin Cao, Panlong Yang, and Shuangshuang Xue. 2018. Control with Gestures: A Hand Gesture Recognition System Using Off-the-Shelf Smartwatch. In *2018 4th International Conference on Big Data Computing and Communications (BIGCOM)*. IEEE, Chicago, IL, 72–77. https://doi.org/10.1109/BIGCOM.2018.00018