RingByte: Enhancing Text-Entry Practicality via A Singular Wearable Rotating Smart Ring

Rucheng Wu
City University of Hong Kong
Hong Kong, China
ruchengwu2-c@my.cityu.edu.hk

Tao Ni
Department of Computer Science
City University of Hong Kong
Hong Kong, China
taoni2-c@my.cityu.edu.hk

Zehua Sun
Department of Computer Science
City University of Hong Kong
Hong Kong, Hong Kong
zehua.sun@my.cityu.edu.hk

Jiande Sun
Shandong Normal University
Jinan, Shandong, China
jiandesun@hotmail.com

Weitao Xu*
Department of Computer Science
City University of Hong Kong
Hong Kong, China
weitaoxu@cityu.edu.hk

Abstract

Mobile text-entry systems play an important role in various smart devices in real-life scenarios, including working, communication, studying, and entertainment. However, existing text-entry methods either require both hands to operate or have specific target operational requirements, making them impractical and prone to privacy leakage for interactive interfaces, smart appliances, and AR/VR scenarios without touch-based keyboards. In this paper, we propose and implement a singular wearable rotating smart ring text-entry system named *RingByte*. The key idea of *RingByte* lies in its rotating interaction design, featuring a rotating outer ring and a stable inner ring, which enables the creation of a novel text-entry method for enhancing practicality. Additionally, we propose a cross-domain attention neural network combined with a weighted allocator to address domain adaptation challenges, accounting for variations in both different individual and different finger characteristics. Through extensive experiments, we demonstrate that *RingByte* achieves user-friendly operation, relatively fast input speed, low error rates, and broad applicability across multiple scenarios. Meanwhile, *RingByte* outperforms four baseline methods and exhibits high accuracy input performance in both cross-person (96.9%) and cross-finger (96.8%) scenarios. These results highlight the excellent practicality, privacy-preserving features, and versatility of *RingByte* as a text-entry interaction system, making it a promising solution for diverse contexts.

CCS Concepts

• **Human-centered computing** → **Ubiquitous and mobile computing**; **Interaction design**.

*Corresponding Author.

UIST '25, Busan, Republic of Korea

ACM ISBN 979-8-4007-2037-6/25/09
https://doi.org/10.1145/3746059.3747717

Keywords

Text Entry; Smart Ring; Domain Adaptation

ACM Reference Format:
Rucheng Wu, Tao Ni, Zehua Sun, Jiande Sun, and Weitao Xu. 2025. RingByte: Enhancing Text-Entry Practicality via A Singular Wearable Rotating Smart Ring. In *The 38th Annual ACM Symposium on User Interface Software and Technology (UIST '25), September 28–October 01, 2025, Busan, Republic of Korea.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3746059.3747717

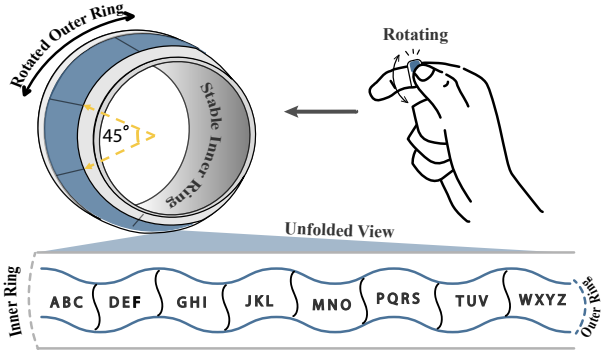


Figure 1: Illustration and architecture of *RingByte* with an unfolded view of the T9 keyboard layout.

1 Introduction

The rapid development of smart interactive technologies, including wearable devices, smart appliances, intelligent projection systems, and augmented/virtual reality devices, has become increasingly prevalent in our daily lives [31, 43]. While these interactive methods have undoubtedly enhanced our quality of life, their text entry capabilities still lag behind those of physical keyboards, particularly noticeable in scenarios including text input in augmented reality environments, command execution for smart appliances, and interaction with wearable devices—contexts where both speed and accuracy are essential. Fortunately, mobile text entry, as one of the most direct and essential interaction methods, has garnered considerable attention, leading to the design and development of

Table 1: Comparison with ring-based text entry systems (○-Low, ◗-Medium,●-High). *RingByte* is the only ring-based text entry system that does not require usage position and finger, is portable, privacy-preserving and practical, based on a ring.

Related Works	One-Ring	Position-Adaptive	Finger-Flexible	Portability	Privacy	Practicality
TypingRing [32]	●	◗	○	◗	○	◗
QwertyRing [15]	●	○	○	◗	◗	◗
TypeAnywhere [51]	○	●	○	○	◗	○
DRG-Keyboard [25]	○	●	○	◗	●	◗
RingVKB [24]	●	○	○	◗	◗	○
WritingRing [17]	●	○	○	●	●	◗
RingByte (Our System)	●	●	●	●	●	●

several innovative text-entry methods based on smartwatch-based gestures [14, 20, 42, 52], capacitive circuits [29], head rotation [50], and ring-based gestures [24, 25, 32, 40].

Despite mobile text-entry methods based on smartwatches, capacitive circuits, or head rotation, while innovative, face several limitations, particularly in terms of cost and the intricacy of the interaction methods. Firstly, the specialized hardware required for these systems, such as advanced sensors for head tracking or smartwatch interfaces—can drive up device costs, making them less affordable for a wide audience. Secondly, the interactive methods themselves can be complex and difficult to master. Users must learn new, non-intuitive operations, which can be physical exertion, especially in head rotation-based methods that may cause neck strain during prolonged use. Therefore, while these methods hold promise, significant improvements are needed to address the cost, usability, and technological challenges. Smart rings, however, could provide a promising alternative. By integrating compact forms, high portability, low-cost sensors, and offering a simpler, more intuitive method of interaction, smart rings can bridge the gap between high-cost devices and practical, user-friendly text-entry systems. However, prior ring-based text-entry methods predominantly employed multiple rings to simulate a physical keyboard, which is impractical and vulnerable to privacy breaches in our everyday settings, such as TypeAnywhere [51], PinchType [11], FingerText [21], and DRG-Keyboard [25]. Although there are single-finger-based approaches like QwertyRing [15] and RingVKB [24], they are not user-friendly as they both need to fix their wrist in one location and require a touch surface to finish one text entry operation. So we ask a question: *does there exist a novel text entry interaction paradigm that has the potential to enhance the accessibility of text entry on mainstream interactive methods within the context of our daily routines?*

To address this query, we propose a creative interaction text entry system named *RingByte*, which uses only one ring, including one low-cost IMU sensor based on the miniature thumb-to-finger interaction mode. Fig. 1 showcases the diagram of *RingByte* with an unfolded view keyboard layout, providing a visual representation of its design and configuration. Precisely, *RingByte* consists of two independent and separate layers: a general wearable ring layer and a rotating processing layer. The general ring, serving as the inner layer, is designed to provide comfortable daily wearability. The outer rotating layer, a ring with embedded IMU, captures time series data to facilitate typing operation recognition and ensure correct character input. The keyboard layout follows the T9 keyboard (9-key input method), which is divided into eight equal parts, each spanning 45 degrees, taking into account both the rotational structure interaction design and the typing accuracy requirement compared with the sequential character layout.

Tab. 1 shows that our system offers several distinctive advantages compared to prior ring-based text-entry systems. Firstly, one-ring means *RingByte* only utilizes an affordable and low-power IMU sensor for text entry. Secondly, *RingByte* is convenient to be deployed in various scenarios as it requires no designated finger, no fixed restriction, and no wearing position requirements, making *RingByte* position-adaptive and finger-flexible. Thirdly, *RingByte* is portable as its size and weight are comparable to that of a regular ring. Lastly, *RingByte*'s privacy and practicality lie in its architectural design and its ability to enable users to directly touch input on the T9 keyboard layout intuitively. However, developing such a low-cost, private, and practical ring-based text-entry system presents the following challenges:

- **Challenge 1: Limited Interactive Surface.** Unlike wristbands and smartwatches, a single smart ring presents a limited typing surface for building a text-entry system.
- **Challenge 2: Continuous Signal Segmentation.** It is necessary to propose a robust typing signal processing method to segment every continuous gesture.
- **Challenge 3: Cross-domain Applications.** Overcoming variations in the same typing signals across fingers, positions, and individuals with consistent recognition results is crucial in designing a position-adaptive, finger-flexible, and person-adaptive system.

RingByte capitalizes on various opportunities to effectively address the aforementioned challenges: *(i)* To realize a single ring text entry system, we have devised a rotating ring with two layers, allowing for random and smooth rotation of the outer layer. This design enhancement significantly improves the privacy and practicality of the system and offers users a highly versatile and user-friendly typing experience. *(ii)* To enhance the segmentation of typing signals, we have developed a novel signal processing technique that enables more accurate segmentation, especially for continuous typing operations. *(iii)* Leveraging the successful application of domain adaptation in the field of IMUs, *RingByte* designs a cross-domain attention mechanism combined with a weighted allocator to mitigate signal variations arising from different fingers, positions, or persons, enabling seamless text input without constraints. By harnessing the power of domain adaptation, our model

Figure 2: Illustration of four COTS smart rings: Samsung Galaxy Ring, Oura Ring, RingConn Ring, and Amovan Ring.

allows *RingByte* to adapt and learn from user-specific patterns and preferences, resulting in enhanced accuracy and personalized typing experiences.

In summary, we present the following contributions:

- **Novel Interactive System.** To our knowledge, *RingByte* is the first text entry system to employ a single ring with a rotating structure, presenting both privacy and practicality.
- **Novel Signal Processing Method.** We introduce a novel typing signal processing method that effectively segments continuous data, which is crucial in further improving the typing accuracy of *RingByte*.
- **Robust Domain Adaptation Model.** We propose a novel model that combines a cross-domain attention mechanism with a weighted allocator for domain adaptation. This model tackles challenges related to position-adaptive interactions, finger-flexible inputs, and variations across users.
- **Comprehensive Evaluations.** We conduct thorough evaluations on a sizable dataset collected with real-world scenarios. The experimental results show that *RingByte* not only has an excellent performance in cross-person and cross-finger situations but also achieves a high input speed of 22.2 WPM and receives a high acceptance.

2 Preliminary Study

2.1 Primer on Smart Ring

Smart rings are compact wearable devices that combine sophisticated technology with elegant design to integrate seamlessly into users' daily routines while delivering a diverse array of intelligent functions. At the core of their architecture are integrated motion sensors, such as *accelerometers* and *gyroscopes*, which are utilized for tracking and interpreting the user's hand movements and operations. Some advanced models incorporate *magnetometers* for enhanced spatial awareness, enabling more nuanced motion tracking and operation recognition. These sensors capture a wide range of motions, from simple finger taps to complex body gestures, enabling the smart ring to execute commands or interact with other smart devices. The device's sleek exterior, typically crafted from premium materials such as aerospace-grade titanium, scratch-resistant ceramics, or medical-grade polymers, serves both aesthetic and functional purposes. It houses a miniature battery, a low-power microcontroller, and wireless connectivity modules like Bluetooth or NFC, allowing it to communicate with smartphones, tablets, or IoT devices. Fig. 2 showcases four commercial off-the-shelf (COTS) smart ring products, highlighting the industry's design diversity and technological progression. The selection includes Samsung's recently launched Galaxy Ring alongside other premium models featuring

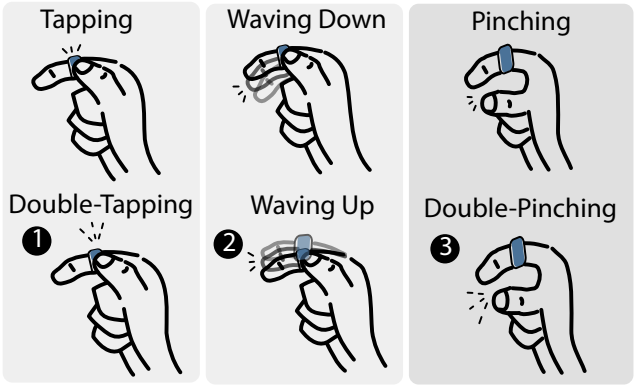


Figure 3: A running example of typing operations in *RingByte* with a single smart ring on the index finger.

distinct material compositions and advanced functionalities. The design emphasizes ergonomic comfort and subtle aesthetics, enabling continuous wear while providing intelligent functionality, including activity monitoring, discreet notifications, and gesture recognition.

2.2 Feasibility of *RingByte*

Unlike existing smart rings limited to basic functionalities like sleep monitoring, step counting, and heart rate measurement, our *RingByte* system implements advanced capabilities, including text entry as a keyboard intermediary. We will now present a detailed feasibility analysis of *RingByte*.

***RingByte* Architecture.** Its double-ring structure we proposed consists of two layers: the inner layer is a general ring similar to those we wear comfortably in our daily life, while the outer layer is a ring with embedded sensors and a battery that can rotate smoothly with the inner ring as the base. The outer ring with embedded IMU incorporates a 3-axis *accelerometer*, a 3-axis *gyroscope*, and a 3-axis *magnetometer* to capture raw data. By synergizing the attributes of both layers, *RingByte* not only maintains the primary function as a stylish adornment but also introduces a new text entry method.

Mapping Angle. To associate the rotation angle with the keyboard character's layout angle, we use the Euler angle (also 360 degrees, corresponding to the rotation plane of the embedded IMU) to establish an absolute mapping relationship between them one by one. Fig. 1 gives the specific distribution of keyboard layout and characters layout in relation to the rotation angle, which is divided into eight equal parts, each covering 45 degrees. This division aligns seamlessly with the renowned T9 keyboard layout, effectively transforming the ring into an intuitive and efficient input device, giving a reasonable solution for the double-ring keyboard.

***RingByte* Typing.** Fig. 3 illustrates a running example that represents the typing operations of the *RingByte* when worn on the index finger. To input a word typing on *RingByte* ring, a user only needs to follow three steps:

- Tap the desired character's zone once you select it after rotating the outer ring.
- Select your desired word by waving ringed finger down.
- Pinch the fingertips of your thumb and finger wearing the ring to confirm your entry word.

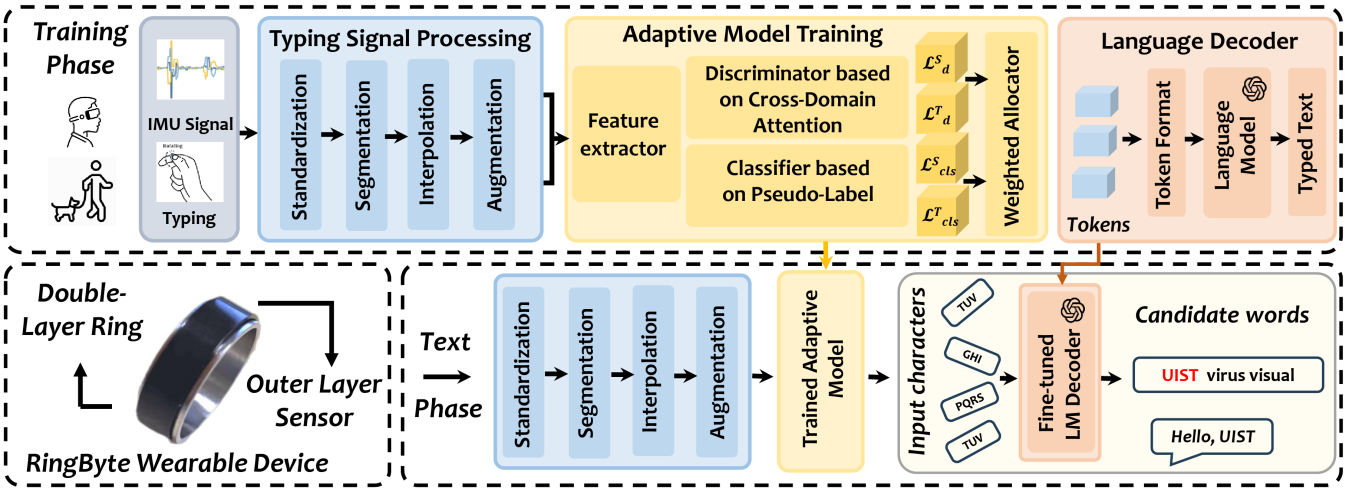


Figure 4: System overview of *RingByte*: 1) Train an adaptive model based on cross-domain attention and pseudo-label methods. 2) Train a language model as text entry decoder. 3) Text entry based on the trained system.

Furthermore, we have integrated three additional gestures into *RingByte* to enhance the overall input experience. These gestures provide convenient shortcuts and actions for users. ❶ Double tapping allows users to effortlessly switch between input languages, punctuation marks, or other symbols, catering to their typing preferences or specific communication needs. ❷ Waving up the finger adorned with the ring slightly serves as a quick and effective way to delete unwanted characters, streamlining the editing process and minimizing errors. ❸ We have implemented another customizable double-pinching gesture that can be programmed to activate specific functions based on the context or user preferences when composing a message, such as screen locking, copy-paste operations, or other application-specific shortcuts. By allowing such specific functions, the *RingByte* provides a flexible and efficient way to interact with the device while maintaining privacy and adapting to different user needs, enhancing productivity and efficiency. In summary, this combination of structural design and subtle, precise interaction allows the system to not only maintain a high level of privacy because of its miniature operations, protecting the user's data from prying eyes in public or shared spaces, but also to provide an intuitive and efficient user experience enhancing its practicality.

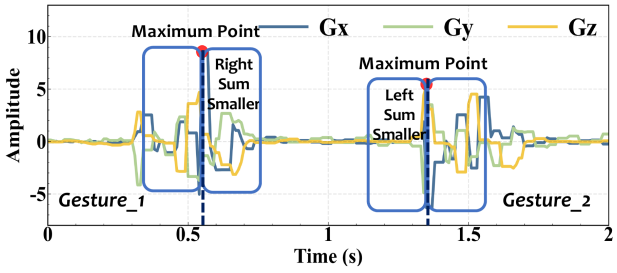


Figure 5: Compare the sum of the twenty data points preceding and following the maximum value point.

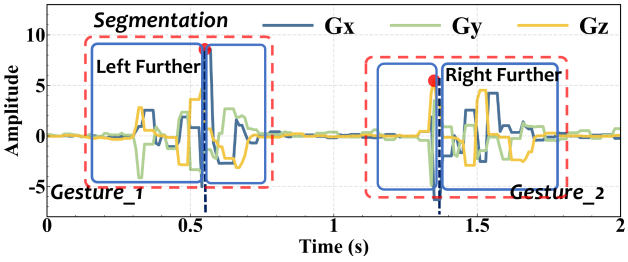


Figure 6: A consistent bimodal gesture segmentation process involving left further or right further scenarios.

3 System Design

3.1 System Overview

The objective of *RingByte* is to develop a text entry system using only one IMU-equipped smart ring and overcome the ring-based text entry limitations of sensitivity to wearing position, dependency on specific finger, the requirement for physical touch surfaces during typing, and degraded performance for new target users. As illustrated in Fig. 4, the system architecture operates through two distinct phases: training and text entry phases.

Training Phase. For the training phase, our raw IMU data after typing signal processing will go into a domain discriminator based on a cross-domain attention mechanism, which facilitates the capture of domain-specific patterns between source and target domains (e.g., a different person or different finger) to calculate the domain losses, and one classifier based on pseudo-label to solve unlabeled target domain problem using the predictions made by the classifier to calculate classification losses, respectively. Furthermore, the losses are inputted into a weighted allocator, which utilizes the losses to calculate sample weights to assign different importance to individual samples based on their relevance between source and target data, guiding the model in prioritizing and leveraging the most relevant and informative data to make accurate typing classification

results. Meanwhile, we trained a language model combining word-level and character-level n-gram methods with autocorrection and edit distance filtering for accurate T9 input prediction.

Text Phase. In the text entry phase, we directly input the raw operation data, captured from the user's specific typing actions, into our system. The data then undergoes signal processing, followed by analysis through a trained adaptive model and fine-tuned language decoder, which processes the input to output the correct text results based on the user's actions.

3.2 Typing Signal Processing

Standardization and Segmentation. Typing data collected from IMU contains 3-axis *acceleration* data, 3-axis *gyroscope data*, and 3-axis *magnetometer* data. Gyroscope data are used to segment gesture signals because of their more distinguishable signal patterns than acceleration and magnetometer data. To make an accurate segment, we first standardized data through z-score [35], which preserves the data distribution and employs the threshold detection method to approximately detect the start point of the gesture. Then, we search for the maximum points in the vicinity of the previously identified start points to refine the determination of gesture start time by eliminating redundant points identified through the threshold method, which has good performance for tapping, waving, and pinching operations, named unimodal typing operations. However, this method is not suitable for typing operations with multiple peaks in continuous operations, such as double-tapping and double-pinching, named bimodal typing operations shown in Fig. 3.

Unimodal Typing. Using the maximum points identified above, we can effectively segment the data by applying an experimentally determined window size of sixty samples. After segmenting the data, we calculate the sum of the first ten and last ten data points in each segment to eliminate offset data. Segments with a calculated sum below the threshold we have set empirically, indicating a notable data deviation, are discarded. By employing this approach, we can refine the segmentation process and ensure that only segments with relatively consistent data are retained for further analysis.

Bimodal Typing. Bimodal segmentation introduces more uncertainty as the maximum point may occur at the second peak, making the segmentation and recognition process more challenging. To ensure the quality of the segmented data, we incorporate an additional condition to calculate the sum of the twenty data points preceding and following the maximum value point identified above simultaneously, respectively. If the sum of the preceding points is less than that of the following points, we will fetch an additional twenty data points backward from the maximum point comparing another side. Conversely, we will fetch an additional twenty data points forward from the maximum point comparing another side. This approach addresses the issue of bimodal peak detection, where the peak may be far from the starting point, which is a common occurrence in continuous typing operation detection. Fig. 5 and Fig. 6 illustrate how to calculate and compare the sum and segment further based on the comparison results. Additionally, we set the window step equal to eighty experimentally, which is larger than sixty of the unimodal segmentation. Once the segmented data is obtained, we employ a method similar to that of unimodal segmentation to calculate the sum of the first ten and last ten data points in each segment to eliminate offset data.

Algorithm 1: Typing Signal Processing of *RingByte*

Input: $\widetilde{\mathcal{A}}$: standardized data with maximum points $\widetilde{a}$; a: standardized data; τ_1: detection threshold; τ_2: filter threshold; l: segmentation length; $\widetilde{\mathcal{G}}$: middle clips.
Output: $\mathcal{G}$: generated signal clips.

for *each maximum point* $\widetilde{a} \in \widetilde{\mathcal{A}}$ **do**
  $sum_1 = sum(\widetilde{a} - 20 : \widetilde{a}), sum_2 = sum(\widetilde{a} : \widetilde{a} + 20)$
  if $sum_1 > sum_2$ **then**
    if $sum(\widetilde{\mathcal{G}}(: 10)) > \tau_2$ *and* $sum(\widetilde{\mathcal{G}}(-10 :)) > \tau_2$ **then**
      $\widetilde{\mathcal{G}} \leftarrow [a(\widetilde{a} - l - 20 : \widetilde{a} + l)]$
    end
  end
  else if $sum(\widetilde{\mathcal{G}}(: 10)) > \tau_2$ *and* $sum(\widetilde{\mathcal{G}}(-10 :)) > \tau_2$ **then**
    $\widetilde{\mathcal{G}} \leftarrow [a(\widetilde{a} - l : \widetilde{a} + l + 20)]$
  end
end
$[G_1, G_2, \ldots, G_n] \leftarrow Interpolation(\widetilde{\mathcal{G}})$
$[G_1, G_2, \ldots, G_n] \leftarrow Generation[G_1, G_2, \ldots, G_n]$
$\mathcal{G} \leftarrow [G_1, G_2, \ldots, G_i]$
Output generated signal clips $\mathcal{G}$.

Interpolation and Augmentation. After obtaining the segmented gyroscope data, we apply the same operations to acceleration data to align the window's starting and ending points. Then, we apply a linear interpolation [4] method to ensure all the segments have the same dimension equal to $6 * 100$, preparing for the domain adaptive neural network. Furthermore, as a result of dividing our rotating plane into eight equal segments, the signal characteristics exhibit variations across these distinct angle zones. Our experimental findings reveal that *the signal pattern of the same gesture remains consistent, with the only difference being the corresponding rotation plane axis, which implies we can use rotation matrix [39] R_a^b to generate the other seven angles' data based on one angle zone data we collected.* According to the rotation plane, we define the rotation matrix as follows:

$$R_a^b = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta & -\sin\beta \\ 0 & \sin\beta & \cos\beta \end{bmatrix}, \tag{1}$$

where β represents the rotation angle compared to the original angle, a means the angle at which we collected data, and b means the target angle we wanted to generate. Additionally, we introduce a random parameter between 0.5 and 1.5 to enhance the diversity of our data, which is multiplied by the rotation matrix, further increasing the range of possible signal patterns. By combining the rotation matrix with this random parameter, we can effectively capture the variations in signal characteristics across different word angle zones, providing a more comprehensive representation of the gestures performed. Alg. 1 illustrates the detailed signal processing workflow of *RingByte*, using the bimodal signal as an example.

3.3 Typing Adaptive Model

This section is divided into three parts to introduce our cross-domain attention in domain adaptive model with weighted allocator (CaDAW): 1) Adaptation problem definition in *RingByte* scenarios, 2) Cross-domain attention discriminator, and 3) Classifier and weighted allocator.

Adaptation Problem Definition. In *RingByte* setting, both cross-person and cross-finger scenarios are considered to prove our system's robustness. Specifically, the collected dataset D contains data from different persons with different fingers, encompassing total C_s classes with label y_i. Subsequently, we separate the dataset into cross-person and cross-finger scenarios. For the cross-person scenario, we sequentially select one user to serve as the target domain, denoted as $D_t^p = \{(x_i)\}$. And remaining portion of the dataset, denoted as $D_s^p = \{(x_i, y_i)\}$, is used as the source domain. Similarly, for the cross-finger scenario, we choose one finger as the target domain in turn, represented as $D_t^f = \{(x_i)\}$, while the rest of the dataset becomes the source domain, denoted as $D_s^f = \{(x_i, y_i)\}$. Meanwhile, we assign a domain label of $d_s = 0$ to the source domain and $d_t = 1$ to the target domain during the training process.

Cross-domain Attention Discriminator. Traditional domain adaptation methods mainly use the CNN network to make discriminators, which is not adequate for complex tasks. We propose a novel cross-domain attention discriminator method based on the self-attention mechanism to solve cross-person and cross-finger scenarios. In terms of the original self-attention mechanism, it consists of three vectors: query Q, key K, and value V getting from the shared query weight W^Q, key weight W^K and value weight W^V [41]. Unlike the self-attention mechanism, our cross-domain attention mechanism incorporates the query vectors from one domain to calculate attention over another domain's key-value pairs enabling a dynamic interaction between the source and target domains, allowing them to exchange information and enhance their respective representations. Specifically, the model employs the target domain's query vector Q_t to calculate attention over the source domain's key-value pairs (K_s, V_s). This allows the model to identify the relevant source features that align with the target query, generating features through the source-to-target domain attention mechanism:

$$Attention(s,t) = Softmax(\frac{Q_s K_t^T}{\sqrt{d}})V_t. \tag{2}$$

where Q, K, and V represent query, key, and value vectors, s means source domain, t means target domain, d represents the dimension of latent features, and T represents a transpose operation. Fig. 7 showcases the detailed inference process of the cross-domain attention mechanism from source to target. Conversely, we can employ the same principle to calculate target-to-source domain attention. By integrating both source-to-target and target-to-source domain attention mechanisms, the model creates a bidirectional interaction that enhances the representation of both domains, improving the overall generalization ability of our CaDAW model.

Classifier and Weighted Allocator. Regarding classifiers, we first use a CNN neural network to extract the latent features in source and target domain data. Meanwhile, features are fed into a fully connected layer to get the predicted label compared with

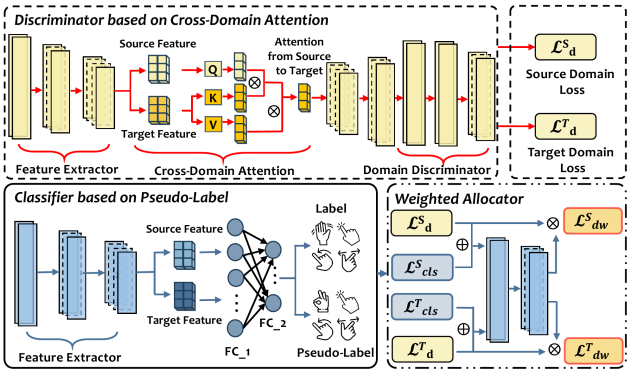


Figure 7: Inference process of cross-domain attention mechanism from source to target domain.

the truth label to calculate source domain classification loss $\mathcal{L}_{cls}^S$. However, the target data is unlabeled when fed into the classifier to improve the model's adaptation ability. Pseudo-label produced by the classifier based on source domain data, therefore, is used to calculate target domain classification loss $\mathcal{L}_{cls}^T$ to train the adaptive model. To mitigate noisy data caused by pseudo-label, we employ a mask and introduce a threshold parameter, denoted as ζ, based on the classification probability P_{pseudo}. Overall, we can calculate the classification loss considering both the accurate labels from the source domain and the reliable pseudo-labels obtained from the target domain to optimize the parameters of the feature extractor and classifier as:

$$\begin{aligned}\mathcal{L}_{cls}(\tau_f, \tau_c) =& \frac{1}{K_S}\sum_{i=1}^{K_S} l_i^c(\tau_f, \tau_c) \\ &+ \frac{1}{\sum_{j=k_S+1}^{k} m_j}\sum_{j=k_S+1}^{k} m_j l_j^c(\tau_f, \tau_c),\end{aligned} \tag{3}$$

where τ_f and τ_c represent the parameter sets corresponding to the feature extractor and classifier, respectively. K_S is the number of samples from training users, k is the total number of samples from training users and new users, and m_j is the mask value of the j-th target sample.

The weighted allocator mainly generates sample weights and outputs two weighted domain losses to optimize the domain discriminator's parameters. The normalized sample weight $w_i(\tau_w)$ obtained from the weight allocator can be calculated through the equation:

$$w_i(\tau_w) = \frac{\varphi_i(\tau_w)}{\sum_{j=1}^{b}\varphi_j(\tau_w) + \sigma(\sum_{j=1}^{b}\varphi_j(\tau_w))}, \tag{4}$$

where τ_w represents the parameter sets of weighted domain allocator, b represents the batch size, and σ is a constant that prevents computation errors when all $\varphi_j(\tau_w)$ are zero. The sample weight, denoted as $\varphi_i(\tau_w)$, can be expressed as:

$$\varphi_i(\tau_w) = W(l_i^c(\tau_f, \tau_c) \oplus l_i^d(\tau_f, \tau_d); \tau_w), \tag{5}$$

where τ_d represents the parameters of the domain discriminator, $\oplus$ represents the concatenation operation, and W is a function that takes $l_i^c(\tau_f, \tau_c)$ as input from the classifier and $l_i^d(\tau_f, \tau_d)$ from

the domain discriminator, with the weighted allocator parameters τ_w. Therefore, the weighted domain allocator loss, denoted as $\mathcal{L}^{dw}(\tau_f, \tau_d; \tau_w)$, can be computed as follows:

$$\mathcal{L}_{dw}(\tau_f, \tau_d; \tau_w) = -\frac{1}{k}\sum_{i=1}^{k} w_i(\tau_w) l_i^d(\tau_f, \tau_d), \tag{6}$$

To summarize, the model's parameters are optimized by considering four main losses listed in Fig. 7: source domain classification loss $\mathcal{L}_{cls}^S$, target domain classification loss $\mathcal{L}_{cls}^T$, source weighted domain loss $\mathcal{L}_{dw}^S$ and target weighted domain loss $\mathcal{L}_{dw}^T$. Based on these optimization processes, a trained classifier based on the pseudo-label combined with a weighted allocator presents robust domain adaptation in both cross-person and cross-finger scenarios.

3.4 Text Entry Decoder

Euler angle and characters layout mapping. After the typing recognition results are obtained, *RingByte* will provide the text content based on the mapping relationship with the character layout angle corresponding to the Euler angle mentioned above. Specifically, we partitioned the Euler angle range into eight equal segments, each spanning 45 degrees. The segments were mapped to corresponding characters as follows: 0°–45° to "ABC," 46°–90° to "DEF," 91°–135° to "GHI," 136°–180° to "JKL," and so on. In addition, users can reconfigure the mapping to their preferred layout, such as QWERTY, because our system utilizes absolute inputs with fixed angle mapping relative to the outer ring. We adopted the T9-based layout because it proved more intuitive and accessible given our novel hardware design and interaction method. To calculate the Euler angle aligning the rotation plane, we utilize the 9-axis IMU to achieve more accurate Euler angle estimation by avoiding static error and error caused by gravity sensors through their complementary capabilities. Based on the rotation relationship between the sensor coordinate system and the geographical coordinate system, we can use direction cosine matrix [26, 36] and Kalman filter [38] to calculate the Euler angle. Specifically, we use quaternion [3] directly according to the equation below:

$$\begin{bmatrix}\varphi\\ \theta\\ \psi\end{bmatrix} = \begin{bmatrix}atan2(2(wx+yz), 1-2(x^2+y^2))\\ arcsin(2(wy-zx))\\ atan2(2(wz+xy), 1-2(y^2+z^2))\end{bmatrix}, \tag{7}$$

where ϕ, θ, and ψ correspond to the roll, pitch, and yaw angles. X, y, z, and w are the quaternion reading from the IMU sensor. Once we obtain the correct Euler angles after rotating the outer ring, we can map them to the corresponding characters according to the keyboard layout. To illustrate this process, let's consider an example where we want to input the word *UIST*. Firstly, we rotate the ring's outer layer to the angle position corresponding to the characters *TUV*. Then, we tap the surface of the outer layer to input the desired characters. Next, we perform similar operations to input characters *GHI*, *PQRS*, and *TUV*. As we input each character zone, our input box will simultaneously display several candidate words for selection based on our text decoder language model. If the first suggested word is not what we intended, we can wave our finger downwards to browse and choose the desired word from the options provided. Finally, to confirm our input, we pinch the fingertips of our thumb and index finger together. By following these steps, we can efficiently input the desired word using *RingByte*.

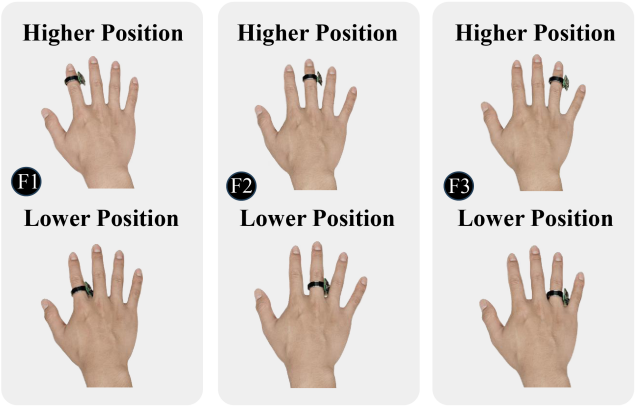


Figure 8: *RingByte* data collection on different fingers and positions: F1 Index finger, F2 Middle finger, F3 Ring finger.

Text decoder language model. To train a language model for a T9 input system, we implemented a hybrid approach combining word-level and character-level n-gram methods based on KenLM [18], followed by an integrated autocorrection process. Initially, we collected a large corpus of T9-compatible text data from NUS SMS Corpus [7] and SMS Spam Collection [1]. These datasets consist of real SMS messages from the era when T9 was prevalent, making them short, informal, and naturally aligned with the T9 keyboard's character set. Then, we standardized the text by converting it to lowercase and filtering out characters beyond the T9 vocabulary (a, 0-9, and basic punctuation like commas and periods). For the character-level model, we trained a 5-gram model to capture fine-grained patterns in keypress sequences (e.g., "ABC", "ABC", "JKL", and "DEF" mapping to "cake"), enabling precise character-by-character predictions. Simultaneously, we trained a 3-gram word-level model using a vocabulary of 100k common English words, leveraging contextual relationships (e.g., "I want to" predicting "bake"). The training data was preprocessed from Wikipedia, ensuring compatibility with the T9 input constraints. For autocorrection, we combined both models: the character-level 5-gram first generates candidate sequences from ambiguous keypresses (e.g., "ABC", "ABC", "JKL", and "DEF" yielding "cake," "bake," "base"), then the word-level 3-gram scores these candidates based on contextual probability (e.g., favoring "bake" in "I want to bake"). An edit distance filter [22] further refines the output by correcting minor keypress errors, ensuring the final result aligns with both statistical likelihood and user intent. This hybrid training and autocorrection pipeline effectively balances prediction accuracy and error correction for T9 inputs.

4 Evaluation 1: Typing Accuracy

4.1 Experimental Setup

Data collection of typing operations. To evaluate *RingByte* typing operation performance, we recruited 21 participants (8 females, 13 males) aged from 20 to 32. Each participant was instructed to perform six typing gestures corresponding to the *RingByte* typing

operations mentioned above. Considering our experimental requirements, we collected data for each gesture 100 times, resulting in 600 times per finger. Meanwhile, we randomly divided the participants into three groups to collect different finger data to evaluate the robustness of our model in both cross-finger and cross-person scenarios. Data was collected using an IMU module (ICM-42607) with a sampling frequency of 100 Hz. It is worth noting that all the data collection was performed on the participants' right hand, and there are no specific limitations on where to wear the ring on the finger collecting data, which is determined by the wearing comfort of different persons. Moreover, the data collection is restricted to the word's angle of collection between 0 and 45 degrees, representing the first character zone. Overall, our data collected not only includes different persons and different fingers but also has no limitations on the wearing positions, shown in Fig. 8.

After completing the data processing outlined above, we obtained gesture data for all eight input zones from 21 individuals and three different fingers. Consequently, we created two datasets to evaluate cross-person and cross-finger scenarios: the *RingByte* cross-person-based (RP-CroP) and the *RingByte* cross-finger-based (RP-CroF) datasets. The RP-CroP dataset comprises 21 individuals, each providing data from two fingers containing approximately 9.6k typing operations. The RP-CroF dataset consists of data from three different fingers, with each finger corresponding to data from 14 different individuals, which contains approximately 67.2k operations of text input. Then, we implement experiments based on these two datasets to evaluate the performance of CaDAW in both cross-person and cross-finger scenarios.

Training Details and Metric 1. We implement the CaDAW model using the PyTorch framework [34] and train the neural network on a desktop equipped with an NVIDIA RTX 3090 GPU featuring 24 GB of memory and an AMD Ryzen Threadripper PRO 3955WX 16-Core processor. For the CaDAW training process, Adam optimizer with an initial learning rate of 0.008 was used for optimization, and the maximum number of training epochs was set to 150. The two datasets' batch size equals 512, and the pseudo-label threshold was set to 0.2 based on our experimental setting. Regarding domain adaptation evaluation in our experiments, we use the leave-one-person-out-cross-validation method with reference to [28]. Meanwhile, we randomly split the source data into training and validation datasets in a 80% to 20% ratio. As for the target data, we split it into adaptive and testing datasets in the ratio of α and $1-\alpha$, respectively, where α takes the values of 0.1, 0.3, 0.5, and 0.8. The model, therefore, trains on the training data, combines adaptive data, tunes on the valuation data, and finally evaluates the target domain testing data. We use three random seeds to report the average performance as the final result for the fairness of the results. We use *Accuracy*, *Precision*, *Recall*, and *F1-Score* as the evaluation metrics that comprehensively evaluate the CaDAW's performance.

4.2 Typing Operation Overall Performance

In the context of *RingByte*, *the precision of typing operations directly reflects the performance of text entry, as each operation corresponds to a specific input command.* Therefore, in the following experiments, we focus on typing operations performance divided into cross-person and cross-finger scenarios, which reflect text entry performance in real-world applications.

Table 2: Performance of CaDAW in cross-person scenario.

Metrics		Accuracy	Precision	Recall	F1 Score
A_{ratio}	**0.1**	96.0%	95.6%	93.4%	94.3%
	0.3	96.3%	95.7%	93.7%	94.6%
	0.5	96.4%	95.0%	93.9%	95.0%
	0.8	96.9%	95.5%	94.7%	94.8%
Overall Average		96.40%	95.45%	94.15%	94.68%

Cross-Person Scenario. To evaluate the convenience of our *RingByte* system for new users, we initially employ the RP-CroP dataset combined with our CaDAW model. The running results for each new user are aggregated to calculate average performance measures. In the RP-CroP dataset comprising 21 users, each individual is considered a new user once, serving as the target user to verify our model's robustness, and the remaining data is utilized as the source data for training. Another significant aspect of our evaluation is the impact of the adaptation ratio size, denoted as A_{ratio}. Therefore, we gradually assess the influence of adaptation ratio size, ranging from 10% to 80%. The comparison results are depicted in Tab. 2, demonstrating our model's robustness even with a minimal adaptation ratio size. Specifically, even with only 10% adaptation ratio size, our model achieves a good accuracy of 96.0%, and it achieves the best accuracy of 96.9% when the A_{ratio} is up to 80%. Additionally, we compared the results without adaptation (A_{ratio} equal to 0) with the results when A_{ratio} equals 10% of the adaptation condition and found that they are both 96.0%. Overall, these results illustrate that our CaDAW model can give satisfactory accuracy to new users even with a slight adaptation ratio size.

Table 3: Performance of CaDAW in cross-finger scenario.

Metrics		Accuracy	Precision	Recall	F1 Score
A_{ratio}	**0.1**	94.9%	93.2%	95.8%	94.5%
	0.3	95.2%	93.7%	96.2%	94.9%
	0.5	96.0%	93.9%	96.6%	95.2%
	0.8	96.8%	94.7%	97.4%	96.1%
Overall Average		95.73%	93.88%	96.50%	95.18%

Cross-Finger Scenario. To further enhance the adaptability of our *RingByte* system and cater to the diverse needs of individuals, we conduct experiments using the RP-CroF dataset to evaluate the robustness of our model in different finger scenarios. Like the cross-person situations discussed earlier, we employ a similar approach, using two out of the three fingers as the source domain and treating the remaining finger as the target domain from the RP-CroF dataset. The results are highly promising, with the average accuracy consistently exceeding 94.9% across varying adaptation ratios. For instance, when the adaptation ratio size is set at 10%, the model achieves an accuracy of 94.9%, which progressively increases to 96.8% when the adaptation ratio size reaches 80%. These findings indicate that even with a smaller proportion of adaptation data, the CaDAW model can adapt effectively to different finger scenarios, showcasing its robustness and versatility. When comparing these results with the performance obtained without adaptation, wherein the accuracy stands at 93.5%, we observe a significant improvement

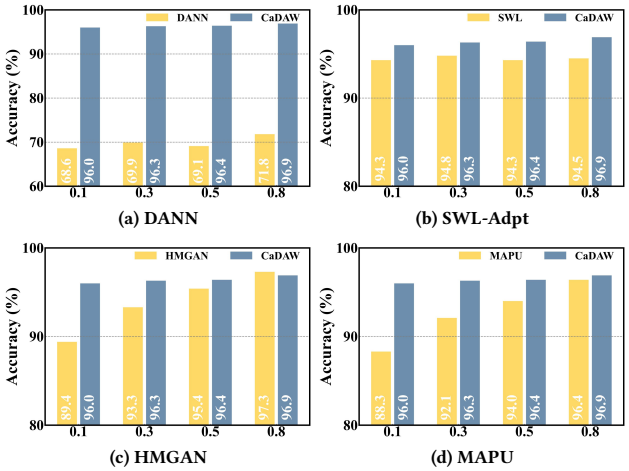


Figure 9: Accuracy comparison performance of CaDAW and four SOTA methods in the cross-person scenario.

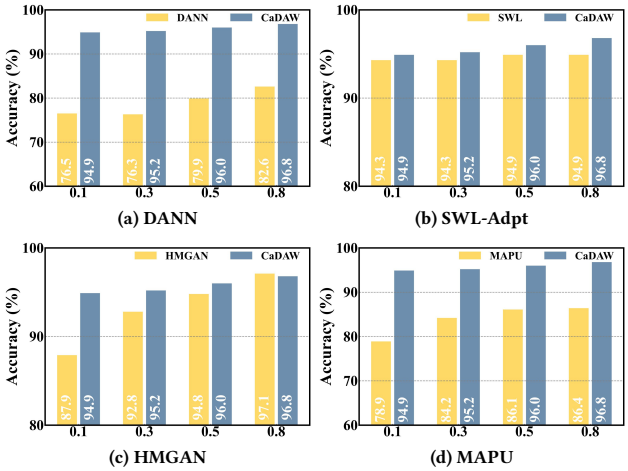


Figure 10: Accuracy comparison performance of CaDAW and four SOTA methods in the cross-finger scenario.

of 3.3% in the cross-finger condition. And the detailed results are shown in Tab. 3. In conclusion, our CaDAW model not only exhibits exceptional performance in addressing cross-person challenges but also demonstrates a remarkable capability to adapt to different finger scenarios.

4.3 Comparison with SOTA Methods

To assess the performance of our CaDAW model, we conduct a comprehensive comparison with several state-of-the-art (SOTA) methods that address the domain adaptation problem. The methods we select for comparison include classic Domain Adversarial Neural Network (DANN) [13], Weighting Allocator-based SWL-Adapt [19], Generative Adversarial Network-based HMGAN [6], and a recent source-free domain adaptation method, MAPU_SFDA [37]. To ensure a fair and consistent comparison, we re-implement all target adaptation settings within our framework while maintaining the same backbone network architecture and training strategies. This ensures that any differences in performance can be attributed to the specific approach employed and not to variations in implementation details. Each of these methods brings its unique approach to the problem:

- **Domain-Adversarial Training of Neural Networks (DANN)** utilizes a gradient reversal layer to train a domain discriminator network to distinguish between domains adversarially.
- **Domain adaptation model with sample weight learning (SWL-Adapt)** employs a parameterized network to calculate sample weights by considering the classification loss and domain discrimination loss.
- **Hierarchical Multi-Modal Generative Adversarial Network (HGMAN)** comprises several modal generators, a hierarchical discriminator, and a classifier.
- **MAsk and imPUte for source-free adaptation (MAPU_SFDA)** randomly masks and captures temporal features in source domains and guides target adaptations.

We thoroughly evaluate the compared methods by calculating their average accuracy, precision, recall, and F1-score values across different adaptive ratio ranges ranging from 10% to 80%. These performance metrics are then compared with our CaDAW model to assess its effectiveness, which is presented in Fig. 9 and Fig. 10. Notably, our CaDAW model achieved an impressive overall accuracy of 96.4% in the cross-person situation and 95.7% in the cross-finger situation, respectively. The classic DANN method achieves an accuracy of only 69.9% and 78.8% for the cross-person and cross-finger situations, respectively, falling significantly short of the required accuracy levels. On the other hand, SWL-Adapt and MAPU demonstrate higher overall accuracies above 90% for the cross-person scenario, yet they still lag behind CaDAW by 1.9% and 3.7%, respectively. In the cross-finger scenario, SWL-Adapt achieves an excellent overall performance of 94.6%, also trailing CaDAW by only 1.1%. However, MAPU's overall performance for the cross-finger scenario is merely 83.9%, which is insufficient for accurate human activity recognition. Furthermore, HMGAN shows exceptional accuracy of 97.3% and 97.1% for the cross-person and cross-finger situations, respectively, when A_{ratio} is set to 80%. These results slightly outperform CaDAW's corresponding accuracy of 96.9% and 96.8%. However, it is worth noting that HMGAN's performance deteriorates significantly as the A_{ratio} decreases gradually from 80% to 10% for both cross-person and cross-finger scenarios. In practical applications, a smaller A_{ratio} indicates better and faster adaptation to new users, making it a critical factor to consider. Consequently, the superior performance and robustness of our CaDAW model, surpassing the compared methods in both scenarios, emphasize its effectiveness and suitability for our *RingByte* system.

4.4 Ablation Study

Below, we present a comparative analysis of the effectiveness of each system module in CaDAW. We design three system variants to assess their impact on typing recognition by removing specific functional modules from the model architecture. To accurately evaluate the effect, we set the value of A_{ratio} to 0.8. The three variants of CaDAW are as follows:

- **CaDAW-D:** This variant is obtained by removing the domain adaptive parts.

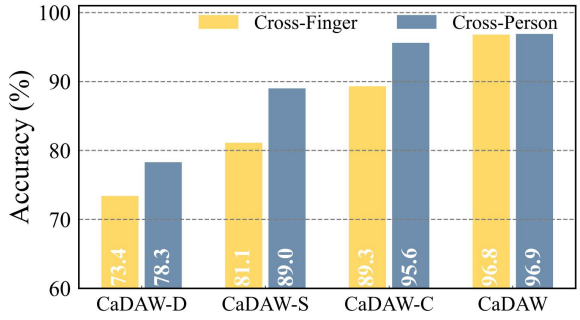


Figure 11: CaDAW model ablation study.

- **CaDAW-S:** This variant is obtained by removing sample weight learning from the target samples.
- **CaDAW-C:** This variant is obtained by removing the cross-domain attention discrimination losses from the input of the weighted allocator.

The results of our analysis are presented in Fig. 11. Firstly, the comparison between CaDAW-D and CaDAW highlights the efficacy of our domain adaptation parts in both cross-finger and cross-person scenarios, with almost 23% improvements in accuracy. Secondly, the comparison between CaDAW-S and CaDAW proves the efficacy of the weighting allocator component when considering target samples in both cross-finger and cross-person scenarios. Finally, the accuracy of variants CaDAW-C and CaDAW demonstrates the importance of the cross-domain attention component, which encourages the system to learn from different domains and enhances its adaptation ability, particularly in scenarios with a limited number of domains, such as the cross-finger scenario.

5 Evaluation 2: Online Text Entry

5.1 Experimental Setup

Prototype. We implement *RingByte* on a custom-built double-layer smart ring, which is based on the DA14583 chipset. It includes a BLE chip and an IMU module (ICM-42607) with a 3-axis accelerometer, a 3-axis gyroscope, and a 3-axis magnetometer chip. The size of the ring is $21mm * 25mm$, and the weight is approximately 12.7g in total, including a 6.4g sensor board and a 6.3g ring structure with a rotation function. In our future work, we will design a better double-layer customized ring with the outer ring embedded with a flexible IMU sensor and well integrated into the outer ring structure. We use the smartphone (rooted OnePlus Ace2 Pro) as our terminal for the edge device to display real-time input character data and typing operations. Fig. 12 (a) shows the prototype scale we used. To provide a clearer illustration of the *RingByte* operation, we affixed white tape to the designated letter regions on the outer ring of the device, as shown in Fig. 12 (b), thereby facilitating participants' understanding of the system's functionality and the corresponding letter-mapping relationships during data collection. These visual aids have no impact on the core operation of the system and are solely intended to help new users get familiar with *RingByte* more quickly; users may freely customize the markings according to their personal preferences, such as applying decorative stickers, engraving symbols, or opting to leave the surface unaltered.

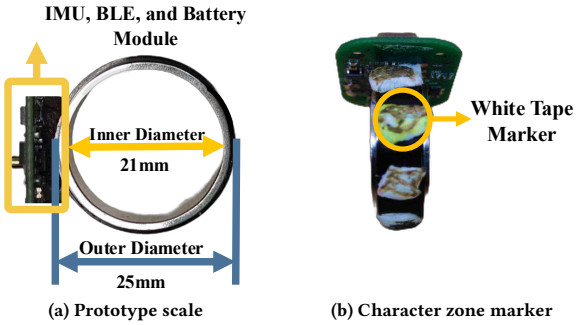


Figure 12: *RingByte* prototype: (a) Scale of *RingByte*. (b) Character zones are marked with white tape.

Data collection of text entry. To evaluate the text entry performance of *RingByte* and our language decoder system under realistic working conditions, we conducted a four-day in-office study simulating daily workplace usage. We recruited 8 additional participants (3 females, 5 males) aged 22-28 to perform text entry tasks in a common office environment designed to simulate everyday computing tasks shown in Fig. 14 (a). Participants self-reported that they were familiar with the T9-based keyboard layout and always used it on their smartphones daily. Each participant was asked to transcribe 20 phrases randomly sampled from the MacKenzie phrase set [30] as quickly and accurately as possible. Before collecting data, there was a 30-minute familiarization phase about the keyboard layout of *RingByte* and the positions of different characters. Each day, participants transcribed 20 phrases in two sessions, with a five-minute rest period after each session. Meanwhile, we evaluated *RingByte*'s performance in different usage habits by having participants input the same phrases using their index, middle, and ring fingers, respectively. Notably, data collection in all three conditions was conducted exclusively on the right index, middle, and ring fingers.

Metric 2: Words Per Minute and Total Error Rate. We assessed the text entry speed using the metric of Words Per Minute (WPM) [2] with this formula:

$$WPM = \frac{|S| - 1}{T} \times 60 \times \frac{1}{5}, \tag{8}$$

where $|S|$ denotes the length of the transcribed string, including blank spaces, and T represents the total time taken to transcribe the string. To evaluate text entry speed under different habits, we initially computed the average speed for three different fingers. Subsequently, we employed these individual results to derive the overall average speed across different individuals, along with the standard deviation (SD), which served as an indicator of the overall performance across different people. Additionally, we reported the Total Error Rate (TER) [47], a metric calculated as the sum of corrected and uncorrected errors in the final output phrases divided by the total number of reference units (e.g., words or characters) through the equation:

$$TER = \frac{IF + INF}{C + INF + IF}, \tag{9}$$

where IF (incorrect-fixed) represents all backspaced characters corrected during text entry, INF (incorrect-not-fixed) denotes all erroneous characters remaining in the final transcription, and C

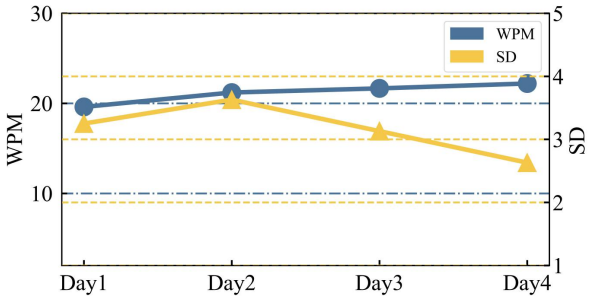


Figure 13: Performance of text entry speed.

(correct) represents all accurately transcribed characters in the output text. TER accounts for insertions, deletions, substitutions, and shifts, providing a comprehensive measure of system accuracy.

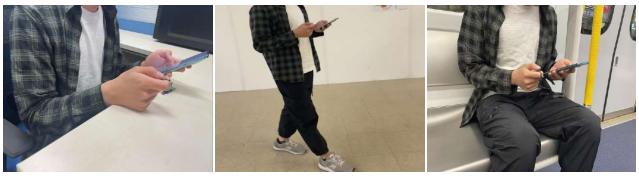
5.2 Text Entry Performance.

Text Entry Speed. Through our experimental scenario setting above, the speeds observed over a span of four days are presented in Fig. 13. The figure illustrates a noticeable increase in text entry speed as the number of days of device usage increases, reaching a peak of 22.2 WPM (SD = 2.63) on the fourth day, which is higher enough compared with ring-based text entry method TipText [49] 13.3 WPM. It is worth noting that even individuals with limited exposure to *RingByte* were able to operate it effortlessly, achieving a text entry speed of 19.60 WPM (SD = 3.25) on the first day, which is also higher enough compared with ring-based text entry method QwertyRing [15] 13.75 WPM. Thus, *RingByte* facilitates an easy learning process with minimal costs. Overall, the speed curve exhibited a non-converging pattern towards the end of the study, which implies that text entry speed may continue to improve with prolonged usage.

Text Error rate. *RingByte* achieved a low total error rate of 2.8% in our experiments, where eight participants completed around 6,400 total trials (20 phrases/day × 10 words/phrase × 4 days × 8 users) over four days, evaluating text entry performance in seated scenarios. This performance was enabled by our optimized rotating hardware design supporting bidirectional rotations (clockwise/counterclockwise) and a robust language decoding system providing real-time feedback that effectively compensated for motion artifacts. Post-study surveys demonstrated around 90% user satisfaction for comfort, with no reported fatigue - confirming *RingByte*'s reliability for extended wearable use.

5.3 Practical Scenario Applications

To comprehensively evaluate the rotating smart ring *RingByte*'s robustness and practical utility, we conducted real-world testing across another two distinct usage scenarios besides the seated scenarios mentioned above: (1) mobile input while walking, assessing performance during physical movement and (2) subway commuting conditions, evaluating functionality in vibration-intensive public transit environments. This multi-scenario approach rigorously verifies *RingByte*'s adaptability to diverse real-life situations while maintaining input accuracy and user comfort. In terms of the latter two scenarios, we adopted an efficient testing protocol by focusing on final-day performance data to validate real-world practicality



(a) Seated (b) Walking (c) Subway

Figure 14: Application scenarios of *RingByte*.

rather than conducting a full four-day longitudinal study. All three scenarios were collected using the same prototype mentioned above, with the ring connected via Bluetooth to a smartphone running our custom input application. Fig. 14 showed the application scenarios of *RingByte*, and the detailed results of each scenario are described below.

Walking Scenario. In the walking condition shown in Fig. 14 (b), participants used the rotating ring input system while walking at a natural pace (1.2-1.4 m/s) along a 30-meter indoor test path without obstacles and turns. The system maintained stable performance with an average text entry speed of 18.6 WPM (83.8% of seated scenario) and a moderate TER increase from 2.6% to 4.2% compared with seated stationary scenarios, demonstrating remarkable robustness to walking interference.

Subway Scenario. The *RingByte* system was tested under real subway conditions during off-peak hours (9:00-10:00 PM) with moderate passenger density shown in Fig. 14 (c). As subways operated at $30 - 50 km/h$ with normal braking stop, the system achieved 18.1 WPM with 3.9% TER - demonstrating intermediate performance between the two other experiment scenarios. Compared to the seated condition (22.2 WPM, 2.6% TER), subway operation showed 18% slower input speed and 33% higher error rates, reflecting the challenges of transit environments. However, it outperformed walking conditions (18.6 WPM, 4.2% TER) with 7% better accuracy and comparable speed. This performance confirms the *RingByte*'s resilience to transit-environment challenges while outperforming mobile walking scenarios in input accuracy.

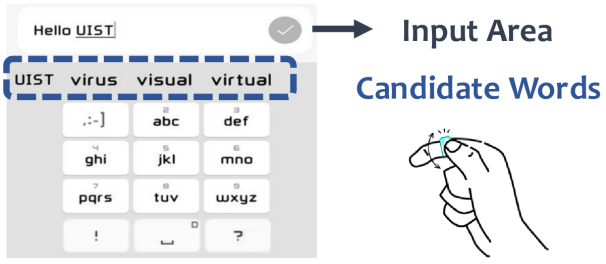


Figure 15: *RingByte* terminal UI.

5.4 Terminal UI

As illustrated in Figure 15, the *RingByte* visual terminal integrates a T9-based keyboard layout, a candidate word selection area, and a dedicated input zone to deliver a seamless and intuitive user experience. The interface is designed to provide clear visual feedback, enhancing usability and engagement. Users navigate character zones by rotating the outer ring, selecting the highlighted character

with a single tap on the ring's surface. For candidate characters or words, a simple downward wave of the ring finger highlights the desired option, offering immediate visual confirmation of the selection. The input is finalized through a precise thumb-to-ring fingertip pinch gesture, which provides visual feedback to ensure confident operation. This multimodal interaction design not only supports efficient single-handed operation but also achieves an impressive typing speed of 22.2 WPM in a seated scenario. The system maintains a recognition accuracy exceeding 96% for typing operations, ensuring reliable performance. The responsive feedback mechanisms—combining visual cues and intuitive gestures—create a fluid and comfortable typing experience, making the interface accessible and efficient for prolonged use.

5.5 User Study

In addition to the aforementioned experiments, we conducted a user study by inviting participants to complete a questionnaire regarding their experience with *RingByte*, assessing it from six different aspects. Each aspect was evaluated using a score ranging from 1 to 5 in Fig. 16, indicating the level of satisfaction with *RingByte*. The first aspect, privacy, has gained widespread acceptance for its effectiveness in privacy protection, with the rotating structural advantages contributing to an average score of 4.30. Portability, another key aspect, plays a crucial role in the overall usage experience. With an impressive average score of 4.28, *RingByte* demonstrates strong potential for seamless integration into users' daily lives. The third aspect, regarding *RingByte* as a replacement for other text entry methods on mobile computers, also received a favorable average score of 4.00. Furthermore, ease of use and the learning effect are identified as important factors that contribute to expanding the range of usage scenarios and user groups for *RingByte*. These aspects received average scores of 4.19 and 3.93, respectively. Lastly, we requested participants to provide an overall score based on their usage experience, as well as considering factors such as size, weight, practicality, and the aforementioned aspects. The majority of participants expressed a preference for *RingByte*, giving an average score of 3.99. They expressed an interest in incorporating *RingByte* into their daily lives and wearing it consistently, as its size is comparable to that of a general ring. These user study findings demonstrate positive feedback and indicate a high level of acceptance and interest in *RingByte* as a practical, privacy-preserving, and wearable text entry solution.

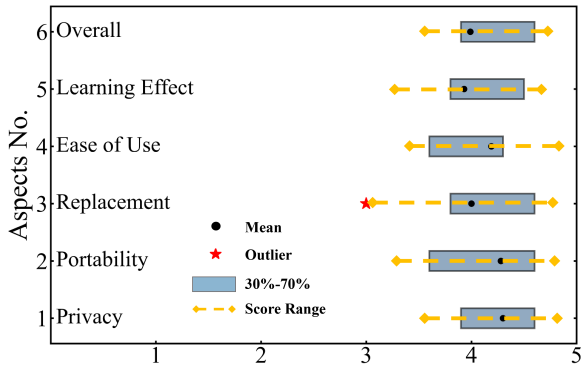


Figure 16: Rating scores results of user study. 1-strongly disagree, 5-strongly agree.

6 Related Work

Text Entry without Physical Keyboard. Virtual keyboards have gained significant attention in various forms, ranging from acoustic, capacitive circuits, and smartwatches to head-mounted operations. MagicInput [33] uses acoustic-based 1D finger tracking technology to realize a text input system. DigiTouch [46], based on glove devices, offers an intuitive and efficient method for inputting text by reimagining the keyboard layout and incorporating thumb-to-finger touch. In terms of smartwatch-based keyboards, a novel approach that utilizes the vibrations captured by an IMU sensor on standard smartwatches to enable precise finger interactions, ViWatch [8], can realize a T9 keyboard writing system with commodity smartwatches. Handpad [29] utilizes capacitive sensing to convert the human hand into a touch-sensitive surface for direct interaction. However, unlike these systems, which are not portable and practical for everyday routines, *RingByte* uses a ring-based design due to its portability, practicality, and compact design.

Ring-based Text Entry. Smart ring serves as an excellent medium for collecting raw data and transmitting it to a terminal, enabling a wide range of inputs and applications [40, 48]. WritingRing [17], which relies on ring-based handwriting mapped to a 2D handwriting dataset for input. RotoSwype [16] tracks wrist rotation for word-level typing. TypingRing [32] allows users to input characters by selecting zones through hand movements and tapping the desired key with their index, middle, or ring finger on the touch surface. TypeAnywhere [51] utilized ten rings to detect finger taps corresponding to specific keyboard positions on the touch surface. Intuitively, it would be more user-friendly and more accessible to operate using fewer fingers to achieve the same functionality. DRG-Keyboard [25] enables a fingertip gesture keyboard using two rings by measuring relative attitude, mapping it to 2D fingertip coordinates, and detecting thumb touch events using combined data. QwertyRing [15] and RingVKB [24] both need to keep their hands in one fixed position to type. Unlike these designs, either needing to fix their wrist in one location or requiring a touch surface, *RingByte* offers a position-adaptive, finger-flexible, one-ring, and privacy-preserving solution to achieve character-level input and ensure user-friendliness.

Domain Adaptation. Domain adaptation has indeed garnered significant attention as a transfer learning method to tackle the domain shift problem [5, 44, 45, 53]. More recently, it has started to be applied to time series data to address the challenge posed by heterogeneous data distributions, particularly in scenarios where deep learning methods exhibit poor performance [10, 19]. The goal of domain adaptation is to leverage labeled data from a source domain and unlabeled data from a target domain to adapt the model's performance to the target domain. To address these challenges comprehensively, we propose a domain adaptive neural network incorporating a weighted allocator combined with a cross-domain attention mechanism to enhance the model's adaptation ability in our text entry *RingByte* system. This approach tackles the domain adaptation obstacles by considering person diversity, finger diversity, and position diversity, all of which are crucial factors to realize the practicality of mobile text entry.

7 General Discussion

7.1 Application Scenarios

The device, consisting of a single ring, has potential applications in various scenarios of our daily lives. Besides, it can serve as a convenient text input medium, easily transitioning from a decorative accessory to a temporary alternative keyboard when traditional input methods are inconvenient. For instance, it is invaluable when walking on the road and needing to respond to a message urgently or while traveling in spaces where physical keyboards are impractical, such as on airplanes or high-speed train seats, and so on. This allows us to effectively handle such situations while maintaining user privacy. Moreover, *due to its one-ring design, it has the potential to serve as an alternative text input method for individuals with limited finger mobility, such as those affected by stroke or amyotrophic lateral sclerosis (ALS)* [12, 27]. To further enhance input speed, iconic characters customized by yourself can be engraved around the outer ring, providing assistance during texting.

7.2 Functional Extension

Beyond text input, the rotating ring *RingByte* can function as an AR/VR controller, allowing users to seamlessly manipulate virtual objects and navigate environments using intuitive rotational and tapping gestures. For professional applications, it can also enable precise presentation control with bidirectional rotation for slide navigation (with speed-sensitive adjustments) and audio engineering tasks like scrolling through mixer channels while pressing to adjust EQ bands. Additionally, we can integrate it with smart home systems, enabling control of lights, appliances, and security systems with simple twists or taps—eliminating the need for multiple remotes or smartphone apps while providing a streamlined, user-friendly interface. Ultimately, the IMU-powered smart ring not only excels in text entry but also unlocks versatile interactions across entertainment, professional, and smart home environments, thanks to its rotational input and compact, wearable design.

7.3 Future Work and Limitations

While *RingByte* enables portable text input through its compact single-ring design, several limitations must be addressed to optimize its usability across diverse scenarios. The current T9-based keyboard layout, while familiar, may not be ideally suited for rotational input, potentially compromising input speed and accuracy. Investigating alternative layouts specifically optimized for circular input could significantly improve both design ergonomics and typing efficiency for a broader user base. To further enhance the user experience, developing an advanced text input decoder leveraging state-of-the-art language models (e.g., BERT [9], GPT-4 [23]) is essential. Such models, which have revolutionized natural language processing, could be fine-tuned to better accommodate the unique constraints and opportunities of rotational input, thereby improving prediction accuracy and fluency. In terms of domain adaptation, the experiments conducted so far have only utilized data collected from the right hand, disregarding the mirror relationship between gestures of the left and right hands. To improve the generalization ability of the model, collecting more diverse data and training a robust model can enable the system to adapt better to different usage scenarios. Finally, realizing the system's applications beyond text input—capitalizing on its portability, compact form factor, and reconfigurable interface—will unlock its full potential.

8 Conclusion

In this paper, we propose *RingByte*, a novel, position-adaptive, finger-flexible, and person-adaptive text entry system by utilizing only one ring with a rotating interaction structure that addresses the practicality and privacy requirements, especially for text entry in interaction with wearable devices, and real-world environment applications—contexts where both speed and accuracy are essential. Additionally, our domain adaptive model, incorporating a cross-domain attention mechanism and a weighted allocator, effectively tackles challenges related to position-adaptive interactions, finger-flexible inputs, and variations across different users and environments. Through comprehensive evaluations of a substantial self-collected dataset, we have demonstrated the robust performance of *RingByte* in cross-person (96.9%) and cross-finger (96.8%) typing operations. Meanwhile, the text entry system *RingByte* can achieve a high average speed of 22.2 WPM and a low total error rate 2.8% across three fingers. By bridging the gap between practicality, privacy, and accessibility, *RingByte* has the potential to significantly improve the text entry experience for a wide range of users.

Acknowledgments

This project was supported by National Key R&D Program of China (Grant No. 2023YFE0208800), the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CityU 11202124 and CityU 11201422), NSF of Guangdong Province (Project No. 2024A1515010192), the Innovation and Technology Commission of Hong Kong (Project No. MHP/072/23), and the CityU MF_EXT Fund (Project No. 9678363).

References

[1] Tiago A. Almeida, José María G. Hidalgo, and Akebo Yamakami. 2011. Contributions to the study of SMS spam filtering: new collection and results. In *Proceedings of the 11th ACM Symposium on Document Engineering* (Mountain View, California, USA) *(DocEng '11)*. Association for Computing Machinery, Mountain View, California, USA, 259–262. https://doi.org/10.1145/2034691.2034742

[2] Ahmed Sabbir Arif and Wolfgang Stuerzlinger. 2009. Analysis of text entry performance metrics. In *2009 IEEE Toronto International Conference Science and Technology for Humanity (TIC-STH)*. IEEE, IEEE, Toronto, Canada, 100–105.

[3] Evandro Bernardes and Stéphane Viollet. 2022. Quaternion to Euler angles conversion: A direct, general and computationally efficient method. *Plos one* 17, 11 (2022), e0276302.

[4] Thierry Blu, Philippe Thévenaz, and Michael Unser. 2004. Linear interpolation revitalized. *IEEE Transactions on Image Processing* 13, 5 (2004), 710–719.

[5] Changhao Chen, Yishu Miao, Chris Xiaoxuan Lu, Linhai Xie, Phil Blunsom, Andrew Markham, and Niki Trigoni. 2019. MotionTransformer: transferring neural inertial tracking between domains. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI'19/IAAI'19/EAAI'19)*. AAAI Press, Honolulu, Hawaii, USA, Article 982, 8 pages. https://doi.org/10.1609/aaai.v33i01.33018009

[6] Ling Chen, Rong Hu, Menghan Wu, and Xin Zhou. 2023. HMGAN: A Hierarchical Multi-Modal Generative Adversarial Network Model for Wearable Human Activity Recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 3 (2023), 1–27.

[7] Tao Chen and Min-Yen Kan. 2013. Creating a live, public short message service corpus: the NUS SMS corpus. *Language Resources and Evaluation* 47 (2013), 299–335.

[8] Wenqiang Chen and John Stankovic. 2022. ViWatch: harness vibrations for finger interactions with commodity smartwatches. In *Proceedings of the 13th*

ACM Wireless of the Students, by the Students, and for the Students Workshop. Association for Computing Machinery, Sydney, Australia, 4–6.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[10] Di Duan, Huanqi Yang, Guohao Lan, Tianxing Li, Xiaohua Jia, and Weitao Xu. 2023. EMGSense: A Low-Effort Self-Supervised Domain Adaptation Framework for EMG Sensing. In *2023 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, Atlanta, USA, 160–170.

[11] Jacqui Fashimpaur, Kenrick Kin, and Matt Longest. 2020. Pinchtype: Text entry for virtual and augmented reality using comfortable thumb to fingertip pinches. In *Extended abstracts of the 2020 CHI conference on human factors in computing systems*. Association for Computing Machinery, Hawai'i, USA, 1–7.

[12] Valery L Feigin, Michael Brainin, Bo Norrving, Sheila Martins, Ralph L Sacco, Werner Hacke, Marc Fisher, Jeyaraj Pandian, and Patrice Lindsay. 2022. World Stroke Organization (WSO): global stroke fact sheet 2022. *International Journal of Stroke* 17, 1 (2022), 18–29.

[13] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *Journal of machine learning research* 17, 59 (2016), 1–35.

[14] Jun Gong, Zheer Xu, Qifan Guo, Teddy Seyed, Xiang'Anthony' Chen, Xiaojun Bi, and Xing-Dong Yang. 2018. Wristext: One-handed text entry on smartwatch using wrist gestures. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, Montreal, Canada, 1–14.

[15] Yizheng Gu, Chun Yu, Zhipeng Li, Zhaoheng Li, Xiaoying Wei, and Yuanchun Shi. 2020. Qwertyring: Text entry on physical surfaces using a ring. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 4 (2020), 1–29.

[16] Aakar Gupta, Cheng Ji, Hui-Shyong Yeo, Aaron Quigley, and Daniel Vogel. 2019. Rotoswype: Word-gesture typing using a ring. In *Proceedings of the 2019 CHI conference on human factors in computing systems*. Association for Computing Machinery, Glasgow, Scotland, 1–12.

[17] Zhe He, Zixuan Wang, Chun Yu, Chengwen Zhang, Xiyuan Shen, and Yuanchun Shi. 2025. WritingRing: Enabling Natural Handwriting Input with a Single IMU Ring. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, Yokohama, Japan, 1–15.

[18] Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Proceedings of the sixth workshop on statistical machine translation*. Association for Computational Linguistics, USA, 187–197.

[19] Rong Hu, Ling Chen, Shenghuan Miao, and Xing Tang. 2023. Swl-adapt: An unsupervised domain adaptation model with sample weight learning for cross-user wearable human activity recognition. In *Proceedings of the AAAI Conference on artificial intelligence*. AAAI Press, WASHINGTON, DC, USA, 6012–6020.

[20] Prerna Khanna, Shirin Feiz, Jian Xu, IV Ramakrishnan, Shubham Jain, Xiaojun Bi, and Aruna Balasubramanian. 2023. AccessWear: Making Smartphone Applications Accessible to Blind Users. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*. Association for Computing Machinery, Madrid, Spain, 1–16.

[21] DoYoung Lee, Jiwan Kim, and Ian Oakley. 2021. Fingertext: Exploring and optimizing performance for wearable, mobile and one-handed typing. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, Yokohama, Japan, 1–15.

[22] Vladimir I Levenshtein et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*. Soviet Union, 707–710.

[23] Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2022. Pretrained language models for text generation: A survey. *arXiv preprint arXiv:2201.05273* (2022).

[24] Zhenjiang Li, Xinglin Zhang, and Chenshu Wu. 2023. RingVKB: A Ring-Shaped Virtual Keyboard Using Low-Cost IMU. *Proceedings of the ACM on Human-Computer Interaction* 7, MHCI (2023), 1–20.

[25] Chen Liang, Chi Hsia, Chun Yu, Yukang Yan, Yuntao Wang, and Yuanchun Shi. 2023. DRG-Keyboard: Enabling Subtle Gesture Typing on the Fingertip with Dual IMU Rings. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 4 (2023), 1–30.

[26] Chen Liang, Chun Yu, Yue Qin, Yuntao Wang, and Yuanchun Shi. 2021. DualRing: Enabling subtle and expressive hand interaction with dual IMU rings. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 3 (2021), 1–27.

[27] M Patrice Lindsay, Bo Norrving, Ralph L Sacco, Michael Brainin, Werner Hacke, Sheila Martins, Jeyaraj Pandian, and Valery Feigin. 2019. World Stroke Organization (WSO): global stroke fact sheet 2019.

[28] Wang Lu, Jindong Wang, Yiqiang Chen, Sinno Jialin Pan, Chunyu Hu, and Xin Qin. 2022. Semantic-discriminative mixup for generalizable sensor-based cross-domain activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 2 (2022), 1–19.

[29] Yu Lu, Dian Ding, Hao Pan, Yijie Li, Juntao Zhou, Yongjian Fu, Yongzhao Zhang, Yi-Chao Chen, and Guangtao Xue. 2024. Handpad: Make your hand an on-the-go writing pad via human capacitance. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, Pittsburgh, PA, USA, 1–16.

[30] I Scott MacKenzie and R William Soukoreff. 2003. Phrase sets for evaluating text entry techniques. In *CHI'03 extended abstracts on Human factors in computing systems*. Association for Computing Machinery, Fort Lauderdale, Florida, USA, 754–755.

[31] Vimal Mollyn and Chris Harrison. 2024. EgoTouch: On-Body Touch Input Using AR/VR Headset Cameras. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, Pittsburgh, PA, USA, 1–11.

[32] Shahriar Nirjon, Jeremy Gummeson, Dan Gelb, and Kyu-Han Kim. 2015. Typingring: A wearable ring platform for text input. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. Association for Computing Machinery, Florence, Italy, 227–239.

[33] Hao Pan, Yi-Chao Chen, Qi Ye, and Guangtao Xue. 2021. Magicinput: Training-free multi-lingual finger input system using data augmentation based on mnists. In *Proceedings of the 20th International Conference on Information Processing in Sensor Networks (co-located with CPS-IoT Week 2021)*. Association for Computing Machinery, NASHVILLE, TENNESSEE USA, 119–131.

[34] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019).

[35] Patrick Perkins and Steffen Heber. 2018. Identification of ribosome pause sites using a z-score based peak detection algorithm. In *2018 IEEE 8th International Conference on Computational Advances in Bio and Medical Sciences (ICCABS)*. IEEE, Las Vegas, NV, USA, 1–6.

[36] William Premerlani and Paul Bizard. 2009. Direction cosine matrix imu: Theory. *Diy Drone: Usa* 1 (2009).

[37] Mohamed Ragab, Emadeldeen Eldele, Min Wu, Chuan-Sheng Foo, Xiaoli Li, and Zhenghua Chen. 2023. Source-free domain adaptation with temporal imputation for time series data. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, Long Beach, California, USA, 1989–1998.

[38] Dan Simon. 2001. Kalman filtering. *Embedded systems programming* 14, 6 (2001), 72–79.

[39] Gregory G Slabaugh. 1999. Computing Euler angles from a rotation matrix. *Retrieved on August* 6, 2000 (1999), 39–63.

[40] Ryo Takahashi, Masaaki Fukumoto, Changyo Han, Takuya Sasatani, Yoshiaki Narusue, and Yoshihiro Kawahara. 2020. TelemetRing: A Batteryless and Wireless Ring-shaped Keyboard using Passive Inductive Telemetry. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, Minneapolis, Minnesota, USA, 1161–1168.

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[42] Keith Vertanen, Dylan Gaines, Crystal Fletcher, Alex M Stanage, Robbie Watling, and Per Ola Kristensson. 2019. VelociWatch: Designing and evaluating a virtual keyboard for the input of challenging text. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, GLASGOW. UK, 1–14.

[43] Anandghan Waghmare, Ishan Chatterjee, Vikram Iyer, and Shwetak Patel. 2024. WatchLink: Enhancing Smartwatches with Sensor Add-Ons via ECG Interface. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, Pittsburgh, PA, USA, 1–13.

[44] Jindong Wang, Yiqiang Chen, Lisha Hu, Xiaohui Peng, and S Yu Philip. 2018. Stratified transfer learning for cross-domain activity recognition. In *2018 IEEE international conference on pervasive computing and communications (PerCom)*. IEEE, Athens, Greece, 1–10.

[45] Hao Wen, Yuanchun Li, Zunshuai Zhang, Shiqi Jiang, Xiaozhou Ye, Ye Ouyang, Yaqin Zhang, and Yunxin Liu. 2023. AdaptiveNet: Post-deployment Neural Architecture Adaptation for Diverse Edge Environments. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*. Association for Computing Machinery, Madrid, Spain, 1–17.

[46] Eric Whitmire, Mohit Jain, Divye Jain, Greg Nelson, Ravi Karkar, Shwetak Patel, and Mayank Goel. 2017. Digitouch: Reconfigurable thumb-to-finger input and text entry on head-mounted displays. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 3 (2017), 1–21.

[47] Jacob O Wobbrock. 2007. Measures of text entry performance. *Text entry systems: Mobility, accessibility, universality* (2007), 47–74.

[48] Weitao Xu, Huanqi Yang, Jiongzhang Chen, Chengwen Luo, Jia Zhang, Yuliang Zhao, and Wen Jung Li. 2024. WashRing: An Energy-Efficient and Highly Accurate Handwashing Monitoring System Via Smart Ring. *IEEE Transactions on Mobile Computing* 23, 1 (2024), 971–984.

[49] Zheer Xu, Pui Chung Wong, Jun Gong, Te-Yen Wu, Aditya Shekhar Nittala, Xiaojun Bi, Jürgen Steimle, Hongbo Fu, Kening Zhu, and Xing-Dong Yang. 2019. Tiptext: Eyes-free text entry on a fingertip keyboard. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, New Orleans, Louisiana, USA, 883–899.

[50] Chun Yu, Yizheng Gu, Zhican Yang, Xin Yi, Hengliang Luo, and Yuanchun Shi. 2017. Tap, dwell or gesture? exploring head-based text entry techniques for hmds. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, Denver, Colorado, USA, 4479–4488.

[51] Mingrui Ray Zhang, Shumin Zhai, and Jacob O Wobbrock. 2022. TypeAnywhere: A QWERTY-Based Text Entry Solution for Ubiquitous Computing. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New Orleans, Louisiana, USA, 1–16.

[52] Hao Zhou, Taiting Lu, Kristina Mckinnie, Joseph Palagano, Kenneth Dehaan, and Mahanth Gowda. 2023. SignQuery: A Natural User Interface and Search Engine for Sign Languages with Wearable Sensors. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*. Association for Computing Machinery, Madrid, Spain, 1–16.

[53] Zhijun Zhou, Yingtian Zhang, Xiaojing Yu, Panlong Yang, Xiang-Yang Li, Jing Zhao, and Hao Zhou. 2020. Xhar: Deep domain adaptation for human activity recognition with smart devices. In *2020 17th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, Como, Italy, 1–9.