IndexPen: Two-Finger Text Input with Millimeter-Wave Radar

HAOWEN WEI*, Worcester Polytechnic Institute, USA
ZIHENG LI*, Columbia University, USA
ALEXANDER D. GALVAN, Worcester Polytechnic Institute, USA
ZHUORAN SU, Worcester Polytechnic Institute, USA
XIAO ZHANG, Worcester Polytechnic Institute, USA
KAVEH PAHLAVAN, Worcester Polytechnic Institute, USA
ERIN T. SOLOVEY, Worcester Polytechnic Institute, USA

In this paper, we introduce *IndexPen*, a novel interaction technique for text input through two-finger in-air micro-gestures, enabling touch-free, effortless, tracking-based interaction, designed to mirror real-world writing. Our system is based on millimeter-wave radar sensing, and does not require instrumentation on the user. *IndexPen* can successfully identify 30 distinct gestures, representing the letters *A-Z*, as well as *Space*, *Backspace*, *Enter*, and a special *Activation* gesture to prevent unintentional input. Additionally, we include a *noise* class to differentiate gesture and non-gesture noise. We present our system design, including the radio frequency (RF) processing pipeline, classification model, and real-time detection algorithms. We further demonstrate our proof-of-concept system with data collected over ten days with five participants yielding 95.89% cross-validation accuracy on 31 classes (including *noise*). Moreover, we explore the learnability and adaptability of our system for real-world text input with 16 participants who are first-time users to *IndexPen* over five sessions. After each session, the pre-trained model from the previous five-user study is calibrated on the data collected so far for a new user through transfer learning. The F-1 score showed an average increase of *9.14%* per session with the calibration, reaching an average of 88.3% on the last session across the 16 users. Meanwhile, we show that the users can type sentences with *IndexPen* at 86.2% accuracy, measured by string similarity. This work builds a foundation and vision for future interaction interfaces that could be enabled with this paradigm.

CCS Concepts: • **Human-centered computing → Human computer interaction (HCI)**; *Haptic devices*; User studies.

Additional Key Words and Phrases: Millimeter wave FMCW radar; Micro-gesture sensing; In-air gestures; Deep Learning; Text input; Cursor interaction

ACM Reference Format:
Haowen Wei, Ziheng Li, Alexander D. Galvan, Zhuoran Su, Xiao Zhang, Kaveh Pahlavan, and Erin T. Solovey. 2022. IndexPen: Two-Finger Text Input with Millimeter-Wave Radar. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 6, 2, Article 79 (June 2022), 39 pages. https://doi.org/10.1145/3534601

*Both authors contributed equally to this research.

Authors' addresses: Haowen Wei, hwei@wpi.edu, Worcester Polytechnic Institute, Worcester, Massachusetts, USA, 01609; Ziheng Li, zl2990@columbia.edu, Columbia University, New York, New York, USA, 10027; Alexander D. Galvan, adgalvan@wpi.edu, Worcester Polytechnic Institute, 100 Institute Road, Worcester, Massachusetts, USA, 01609; Zhuoran Su, zsu2@wpi.edu, Worcester Polytechnic Institute, 100 Institute Road, Worcester, Massachusetts, USA, 01609; Xiao Zhang, xzhang25@wpi.edu, Worcester Polytechnic Institute, 100 Institute Road, Worcester, Massachusetts, USA, 01609; Kaveh Pahlavan, kaveh@wpi.edu, Worcester Polytechnic Institute, 100 Institute Road, Worcester, Massachusetts, USA, 01609; Erin T. Solovey, esolovey@wpi.edu, Worcester Polytechnic Institute, 100 Institute Road, Worcester, Massachusetts, USA, 01609.

2474-9567/2022/6-ART79 $15.00
https://doi.org/10.1145/3534601

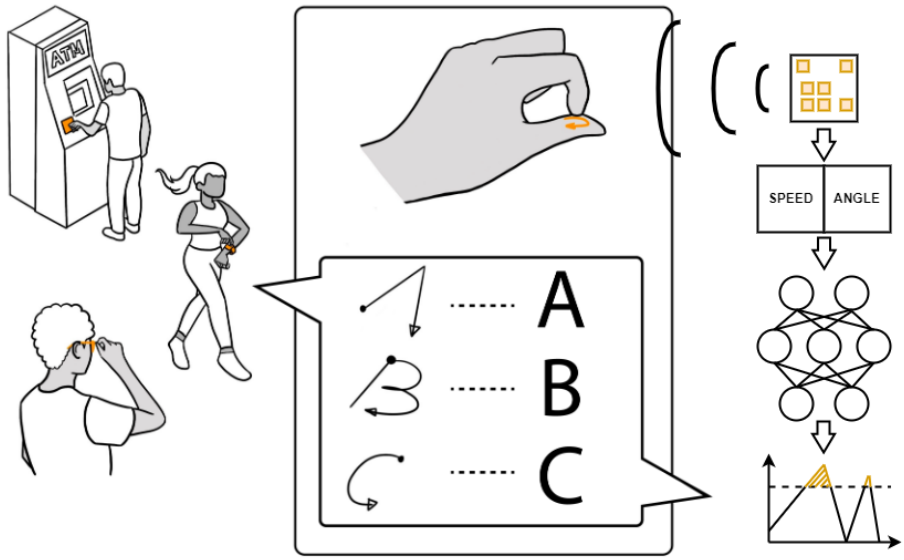

Fig. 1. *IndexPen* enables in-air two-finger text input on devices with limited physical size and no instrumentation required to be worn on the hand. For example, it could provide a contact-free typing interface on public devices such as ATMs to improve hygiene. It provides an alternative to typing on a small screen where typing is challenging and voice input is not appropriate or feasible. *IndexPen* recognizes gestures akin to handwriting including 30 classes. The system processes the speed and angular profile generated by a miniature radar device and uses neural networks to detect the gestures.

1 INTRODUCTION

Recent work has shown that millimeter-wave (mmWave) radar sensors, which are becoming viable in a small form factor, can track finger motion in a 3-dimensional space, opening new doors for human-computer interaction. In particular, this technology could allow for gesture recognition as input to systems, when other modalities are not practical (Figure 1). For example, providing text input on a smartwatch is challenging due to the small size, and voice input is not always convenient or appropriate in certain contexts. In addition, radar-based input has potential to work through materials, so it could enable simple input without requiring a device to be taken out of a pocket. Further, because it is touch-free, it could provide more hygienic interactions in public spaces and medical settings.

With the recent integration of radar sensors into commercial mobile devices (e.g. Google's Pixel 4), we envision an increasing use of such sensors in new contexts. To realize this potential, there are several technical challenges that must be overcome and considerations that are specific to radar-based user interfaces. Most previous work using radar sensors only support a limited vocabulary [3, 55, 61], and do not fully exploit the complex features available from mmWave radars.

In this paper, we make a systems contribution showing a novel prototype that supports text entry and has the potential to work in ubiquitous settings, which is important in wearable interaction, augmented and virtual reality, and other reality-based interaction paradigms [23]. *IndexPen* takes advantage of mmWave radar to recognize micro-gestures performed with two fingers as the sole mode of interaction, and without requiring any instrumentation on the user. *IndexPen* is aimed at extending desktop interaction experiences to smaller devices

with limited processing power. The proof-of-concept system demonstrates novel human-computer interaction that goes beyond existing mmWave radar user interfaces.

Our main contributions are as follows:

(1) We present the design and functionality of *IndexPen*, which enables users to input text by writing the English alphabet with the index finger on the face of the thumb. It is intended to reproduce the feeling of writing with tangible instruments. To achieve an acceptable typing experience and combat the static noise, the gesture set includes 30 distinct symbols (26 letters and additional utility characters including *space*, *backspace*, *enter*, *activation*). The *activation* gesture is designed to activate and deactivate the *IndexPen* and avoid unintended input, and we also include *noise* data to differentiate the gestures and static noise.
(2) We detail the real-time processing pipeline we have developed to support the *IndexPen* interaction paradigms. It utilizes two major feature sets from mmWave radar. The model takes these features as mixed input and uses convolutional recurrent neural networks (CRNN) to resolve what gesture is being performed. The predicted probability from the neural networks is processed with a debouncing algorithm to yield keyboard-like character input. We also report experiment results with a clutter removal algorithm which significantly increases the robustness of the model and shows that the *IndexPen* approach is able to distinguish 30 *IndexPen* symbols with a high 10-fold cross-validation accuracy of 95.89% over 31,000 gesture samples.
(3) We investigate considerations for practical real-world usage of *IndexPen* with new users writing full sentences. We demonstrate the generalizability of *IndexPen* to new users through a series of experiment with user-specific calibration based on transfer learning. After adapting to new users' writing with 20 samples per class in the leave-one-out experiment, the model achieved 87.55% accuracy across five users. We further investigate the learnability of designed gestures and the robustness of the *IndexPen* pipeline in real-world text input scenario with 16 participants over five sessions on separate days and collect their suggestions and feedback for *IndexPen* through the experiment process. We analyzed both factors through objective quantitative results and subjective participant feedback, providing valuable analytical results to other researchers in gesture motion detection area. The average F-1 score across over all participants and the average string similarity [19] in the last session is 0.8815 and 0.8619. Our work addressed the undefined variance between individuals [55] in gesture motion detection.

We also share our compiled data, including the video recording for participants' hands, and the source code for the radar firmware and the data collection software, and the script that processes the data and provides the evaluation results. Additionally, the instruction software (Figure 19), data recording interface (Figure 20), and real-time inference interface (Figure 21) can extend to other HCI studies and benefit the community. The detailed description can be found in Appendix A.3 [1].

2 RELATED WORK

Our work builds on previous work on micro-gesture recognition as well as on radar-based user interfaces. We provide background on these in the sections below. We also discuss the broader category of radio frequency-based signals that can be used in HCI.

2.1 Micro-gestures in HCI

With the increase in micro-gestures in HCI, there have been numerous types of gestures proposed. To gain insight into end-user preferences, Chan et al. [7] conducted a user elicitation study of single-hand micro-gestures, and we refer the reader to this paper for an in-depth exploration. Their study identified four main categories of micro-gestures: *tap*, *swipe*, *draw*, and *circle*. They also found that out of the four, *taps* and *swipes* were the most commonly used by end-users.

[1] https://github.com/ApocalyVec/IndexPenDemo

The *Draw* was not frequently used by end-users but is the basis of the *IndexPen* interaction we present. Because we propose an alphabet resembling handwriting, the draw gestures can be easily remembered, which may have been an issue for other *draw* gestures. Due to the fact that the thumb can comfortably reach the other parts of the hand, Chat et al. found that the thumb was used more frequently in general, and was used on all the elicited swipe gestures performed by the participants [7]. In *IndexPen*, we combine the thumb and the index finger.

2.2 Form Factors for Enabling Microgestures

Different form factors for enabling micro-gestures in the hand have been explored. These can be categorized by whether they take an on-body or an in-air approach.

2.2.1 On-body. On-body devices have been proposed for micro-gesture detection of the hand as they can capture physical or physiological data well through direct contact with the user's skin. *TipText* [66] is a miniature QWERTY keyboard broken into a grid with several letters in each section. The physical device is a conductive flexible printed circuit wrapped around the tip of the index finger, sensitive to touch. By clicking the tip of the index finger with the thumb, users can select a grid section and the specific letter within that section is disambiguated using a language model. Benefiting from the small size, *TipText* has the potential to be integrated with watches and smartphones. Electromyography (EMG), which senses muscle activation during motor movements of the hand and fingers, provides another way to detect gestures by directly decoding the muscle activities [50, 51]. EMG sensors are usually taped to the skin to receive a direct reading of the electrical fields. Unlike mmWave sensing, the user is required to have application-specific sensors attached to their body. When the sensor is mounted to areas such as the tips of the fingers, it can hinder the user's ability to perform other tasks such as typing. Force-sensitive resistors were used in *WristFlex* to successfully recognize subtle finger pinch gestures [13]. *FingerPad* [8] has the most similar interaction properties to *IndexPen* but uses two nail-mounted devices that detect magnetic field intensities and transforms that to coordinates for a user interface.

2.2.2 In-air. Compared to on-body methods, in-air approaches, such as *IndexPen*, for gesture detection may be more favorable as they do not require the user to attach any electronics to the body. This removes some of the device-specific constraints and can present advantages on aspects of sanitation, aesthetics, and utility.

For micro-gesture detection, many optical methods are used to extract the features of the user. A 3-D time-of-flight (ToF) camera [28] has been used to simplify segmentation between the hand and arm. In using a time-of-flight camera, the depth features are included to distinguish certain gestures. Some gestures may have the same projections if captured by a 2D camera. Depth information has the advantage of providing another rich set of features about the distance between the sensor and the parts of the hand while gesturing. However, the ToF camera has disadvantages comparing to 2D camera and that is its low spatial resolution. Decoding fine finger movements often requires millimeter-level accuracy at close range [61], which is hard for the ToF camera to identify dynamics at these small scales. Marin et al. demonstrated the effectiveness of using the Kinect camera in human body recognition [39]. They first extracted the gestures from the acquired depth and color data and then two different types of features were computed from the 3D points corresponding to the hand. They also explored Leap Motion's optical method and reported that it differs from the depth camera which returns a complete depth map. The LeapMotion provides a higher level but more limited data description while Kinect provides the full depth map. LeapMotion provides the coordinates of the major joints of the hands [63]. However, as a camera based approach, the LeapMotion uses approximation when the finger overlaps [26], making it harder to discriminate micro-finger gestures as their design often involves touching between fingers [16, 61, 66]. On the other hand, the velocity information obtained by mmWave sensors can be advantageous in this case as it shows the fine dynamics of the fingers moving at different speeds [22].

In addition to camera-based 3D tracking, magnetic sensing [9] allows precise multiple fingertip tracking. It requires users to attach electromagnet on the tracking target. SkinTrack [70] treats the skin on the arm as a 2D touch surface that can detect hover and touch events. It is enabled by an RF pulse emitter attached to the gesturing finger and an electrode band worn on the wrist. By contrast, mmWave radar sensing does not require any additional equipment on user's body, making it possible to be used as a universal input modality such as an elevator button or on ATM.

Larger gestures have also been explored in air, without instrumentation. Researchers have explored mouse control via nose or head gestures [15, 60], in ubiquitous settings without on-body sensors. These systems has lower resolution since they are based on the movement of the head, and also can be inconvenient in some settings. Other hands-based in-air input methods leverage large-scale gestures and cannot reach high input resolution because individuals may move their hands in different ways [1].

2.3 Enabling Interactions with Millimeter-Wave Radar and Other Radio Frequency Cloud Data

With *IndexPen*, we aim to capture in-air micro-gestures and control the coordinates in a mobile device accurately and conveniently. To do this, we use millimeter-wave radar, which is one type of radio frequency (RF) information that is increasingly available for human-computer interaction applications [42]. Radio-frequency (RF) data from existing infrastructure (e.g. WiFi, Radio-frequency identification (RFID), unmodified global system for mobile communications (GSM)) have been explored to recognize human activities [20, 59], locations [59], and gestures [38, 46, 71, 72]. This is accelerating as 5G standards emerge, with devices possessing increased speed, reduced latency, and increased energy efficiency, and lower cost [42].

Millimeter-wave radar's frequency-modulated continuous-wave (FMCW) transmits a continuous wave modulated in a frequency range, capturing spatial and temporal information of objects. The data profile has a high signal frequency and established processing pipelines [18, 22, 31, 32, 49, 55, 68], making possible sub-millimeter accuracy. In addition, its compact size, low computational cost, and low monetary cost have led to increased interest and investment [3, 22, 31].

Unlike WiFi-based gesture applications, mmWave radar is a miniature portable device and is not constrained by the location of the fixed infrastructure (e.g., WiFi router). Having higher spatial and temporal resolution [22], mmWave sensors can go beyond detecting large-scale gestures (WiFi and camera-based) to sense micro-gestures. Moreover, unlike computer-vision approaches to gesture recognition, radar user interfaces are not impacted by ambient light [39], do not require line of sight [26, 63] and are only marginally affected by temperature comparing to infrared sensing [16]. In addition, radar information contains dynamic profiles that reflect the distance, angle, and velocity of objects, which can be used for robust gesture detection, with shallower processing pipelines [61]. Despite the promise of mmWave for interaction, there are some limitations [30]. mmWave radars lack spatial resolution compared to capacitive or optical sensors, making it hard to distinguish similar gestures when moving fingers reside in close proximity with each other. Further, unlike camera image data, radar data can vary across different platforms in their size and resolution, which calls for domain-specific models. Real-time gesture recognition can also be difficult due to the high throughput, requiring solutions with less complex processing pipelines, often sacrificing some accuracy. Finally, because the features are unique compared to other sensing technologies, it is not trivial to adapt existing pre-processing and prediction methods. The lack of distinct and human-readable features also pose difficulties in visualizing the data and creating better models.

With these considerations in mind, radar-based applications have been explored in diverse contexts. Object detection has been shown with applications in automatic waste sorting, support for the visually impaired, and medical uses [67]. In-car infotainment control has also been explored with radar-based user interfaces [55]. Smart house control via mmWave sensing [32] allows users to control indoor devices remotely. *Pantomime* [43] demonstrated robust recognition of 21 mid-air gestures using mmWave radar in several different indoor

environments. Unlike *IndexPen* which explores two-finger microgestures, these gestures were large hand and arm motions. Other work has demonstrated a gesture recognition pipeline for mmWave data for cursor controls [30] as well as the recognition of eleven gestures [31, 55], resulting in the creation of post-processing algorithms [18, 32, 49] to enable real-time interaction. Our work with *IndexPen* goes further to explore the classification of 31 classes. In addition, the design of the *IndexPen* gestures facilitate the natural performance of the finger motions that mimic the paper-and-pen writing experience. The composition of the *IndexPen* set is further explained in Section 3.1.

2.4 Deep Learning in Interaction Technologies

Another challenge that novel sensing technologies pose is the complexity of data produced by the sensor. Unlike the more mature classes of input devices such as traditional keyboard, mouse, or capacitive sensing, devices utilizing electromagnetic waves propagated in space generate high dimensional data, both spatially and temporally [16, 30, 32, 45, 63]. To overcome the challenges, researchers look to machine learning algorithms to interpret the interaction. Among them, deep learning has shown much prominence lately through the use of neural networks centered on image-based gesture systems [4, 41]. Recent development in radar-based systems have started to explore this direction as the profile from mmWave sensors can be interpreted graphically (see Section 4.1) [36, 61, 68]. At the same time, past research applying deep learning in the image-processing realm have shown that immense datasets are needed for building reliable systems. Even when a dataset is extensive given the application, the models tend to suffer from heavy over-fitting, rendering the system unusable in practice [40]. Additionally, pre-trained models usually do not work well on new users [12], being over-fitted to the user group from whom the training samples are collected. Liu et al. [33] take steps toward a user-independent system by extracting the overall motion trajectory as a high-level feature to minimize the variance between users. The algorithm calculates the centripetal movement energy and centrifugal movement energy for the entire motion. The study demonstrated elevated robustness of the model for new users on five gestures.

Other domains with highly complex and user-specific signals, such as brain-computer interfaces, have faced similar challenges. Widely-used methods to improve the robustness and generalizability of the model are to add weight penalties during training; it encourages the model to learn at every learning step, therefore stabilizing the model's accuracy/loss on validation samples [34, 69]. The user-specificity issue can be mediated with transfer learning, and real-time calibration [14, 25]. Some recent gesture-related work indeed leverage these techniques [24]. We later present a detailed analysis to show the effectiveness of these methods in the first and second parts of the user studies in Sections 5 and 6.

3 INDEXPEN INTERACTION TECHNIQUE

IndexPen's gestures are all performed with a single hand, using the same hardware. In this section, we provide an overview of the interaction model and the pipeline for radar signal processing.

3.1 IndexPen Overview

IndexPen enables text entry as gestures performed by the index finger writing on the face of the thumb. To minimize additional training for the user, we aimed to create an input mode to reflect a person naturally writing on a paper with a pencil. The movements of the fingers reflect the strokes of hand-written characters. This is easy for the system to learn and recognize. We took inspiration from the Palm Pilot alphabet [48]. This writing table was developed in the early days of handwriting recognition, and therefore the strokes are designed so that they can be easily and correctly interpreted. However, the PalmPilot alphabet was designed for hand-held writing pads and the features used are the pixel values. For *IndexPen*, the features are based on the dynamic profiles produced by mmWave radars.

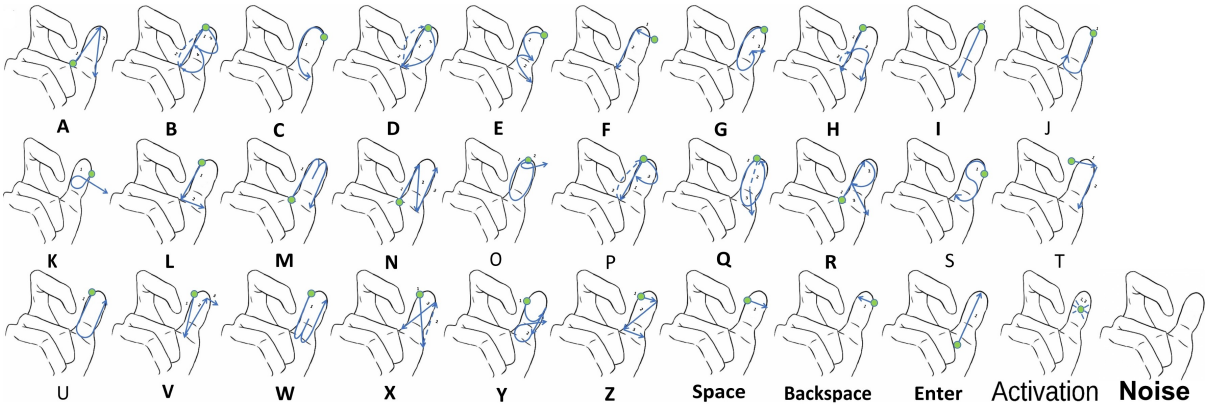

Fig. 2. The *IndexPen* Gesture Alphabet. The basic text-entry interface has 26 letters, a delete, space, enter, activation, and a noise. The illustration's gesture is with the right hand on which the green dot represents the starting position for the index finger on the thumb.

3.2 IndexPen Alphabet

Based on these design considerations, we created the *IndexPen* gesture alphabet (Figure 2). It includes 30 fingertip writing gestures, representing 26 letters and four utility keys. While striving for natural movements, we also take into account whether such gestures are distinguishable through the mmWave radar features. As will be described below in the Radar Signal Processing section (Section 4.1), the radar detects the range, velocity, and angle of the objects. Letters with similar starting points, traces, and endpoints are likely to be confused. Based on these considerations, the letters such as A, F, K, and T are simplified into one-stroke, while letters including O and V are given embellishments to make them more distinguishable from C and U respectively (Figure 2).

In addition to the 26 English letters, we add four utility characters to the table, including *space*, *delete*, *enter*, and *activation*. These characters were designed with an emphasis on the ease of performing the gestures and the likeness to paper-and-pen writing experiences. The *space* gesture is a slide towards the directionality of natural writing (to the right of the written character), whereas the *delete* gesture is the reverse of the *space*. *Enter* is a forward swap intended to mimic pressing enter to send the message action.

Interaction in a noisy real-world environment can introduce challenges, with unintended gestures being a prime concern. This is particularly relevant for in-air interaction, which is not restricted to a specific interaction area (as compared to a touchscreen that is limited to the 2D surface). *Gesture spotting* is the detection of whether there is a gesture or not. A low false-negative rate for gesture spotting is essential for avoiding accidental input. To address this issue, we added a special gesture class that is not a keyboard key - *Activation*. The *Activation* gesture is a quick double-tap between the tip of the index finger and the thumb. This gesture was designed to have high distinguishability amidst a noisy surrounding in our preliminary studies. The use for this gesture is to activate/deactivate an input sequence to *IndexPen*; that is, to use the device, the user first performs the *Activation* gesture. Spotting the gesture, *IndexPen* starts to detect the other 30 text-related gestures. To end the input sequence, the user would perform *Activation* again to deactivate *IndexPen*. Additionally, to distinguish the static noise and gesture input, we include a *Noise* class during our data collection in which participants keep their hands stationary in front of the radar.

4 INDEXPEN ARCHITECTURE

4.1 Hardware Front-end

Our system is based on the Texas Instrument's IWR6843AoP (antenna on the package), an integrated compact mmWave radar device. It operates between 60 to 64 GHz thus giving the best range resolution of around 3.75*cm*, well-suited for micro-gesture detection applications.

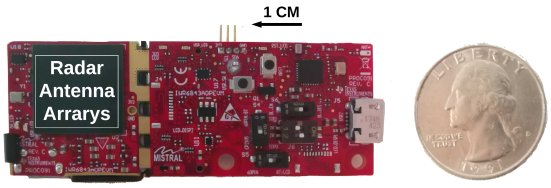

Fig. 3. The physical appearance of the IWR6843 Antenna-on-Package radar device, and its comparative size. The actual radar piece (encircle with the white square) is much smaller than the development board (the red part) where it sits.

4.1.1 Radar Abstraction of Gestures. FMCW mmWave Radar can detect the distance, the radial velocity, and the angle of arrival (AoA) of objects relative to the sensor. The gesture recognition pipeline uses this information to detect small movements of the finger.

The *radial velocity* is one of the keys for gesture detection, as the fingers usually reside within the same range, but during dynamic gestures, fingers typically move at different speeds relative to each other. This creates a dynamic profile that provides a fertile feature space for micro-gesture detection. We also note the velocity is sometimes referred to as *doppler* as the velocity values are linearly mapped from the doppler spread spectrum of the radar signal.

The radar's perceptive power is defined by the range and velocity resolutions given by the following formulas:

$$R_{res} = \frac{C}{2 \times F_B} \tag{1}$$

where F_B is the frequency band where the mmWave radar operates. C is the light speed (3×10^8) m/s.

$$V_{res} = \frac{\lambda}{2T_f} \tag{2}$$

where λ is the wavelength for the starting frequency of the radar operating frequency band. T_f is frame duration dictated by the frame per second (FPS). So a trade-off is shown here between either getting higher velocity resolution or higher frame-rate (shorter frame duration).

In addition to the *range* and *radial velocity*, we must also handle the situation when fingers' movements move within a unit of range and velocity resolution (i.e., fingers move at the same speed close to each other). The radar sensor can also provide angle information, a piece of valuable information for the micro-gesture detection to determine the geometry of the gesture performing hand. The angle information is calculated through the antenna array on the radar device. Similar to the resolutions of range and velocity, the angle is given by:

$$\theta_{res} = \frac{\lambda}{N \times d \times cos(\theta)} \tag{3}$$

where d is the distance between individual antennas in the antenna array, and θ is the angle of objects. Equation 3 is non-linear. The angle resolution is the best when θ is at zero degrees or when the hand is along the central axis of the radar.

The resolution equations (Equations 1, 2, and 3) above build a foundation for devising a suitable signal shape for micro-gesture recognition. By using Equations 1 and 2, our parameters for the radar configuration give $r_{res} = 4.4cm, v_{res} = 0.16m/s$.

This high range and velocity resolution allow for the detection of fine finger motions. Moreover, the non-linearity of the angle resolution (best when along the central axis) advises that the gesture needs to be performed directly in front of the radar. In this work, we made an empirical decision as to have the best range and velocity resolutions. Future studies can investigate how the parameters and resolutions affect the perceptive power of mmWave radars in resolving micro-gestures.

After each radar signaling cycle, a radar frame is packaged and sent to the next stage of the pipeline. Each frame includes two dynamic profiles: the **range-doppler** (RD) and the **range-azimuth-elevation** (RAE), with information of the gesture performing hand's range, velocity, and angle. These are detailed below.

4.1.2 Range-Doppler Profile. The range-doppler profile is obtained at the end of the 2D Fast Fourier Transform (FFT) (refer to [22] for full detail of the radar RF processing pipeline), which represents the range and velocity information of the objects in front of the radar. The profile is a two-dimensional matrix with columns being the velocity and rows being the range. At each point in the matrix, the value is the amplitude calculated through the 2D FFT and represents how strong the reflecting signal is at a specific range and moving at some velocity. It can help to visualize the features as heatmaps. Figure 4 shows how motions are reflected in the range-doppler profile.

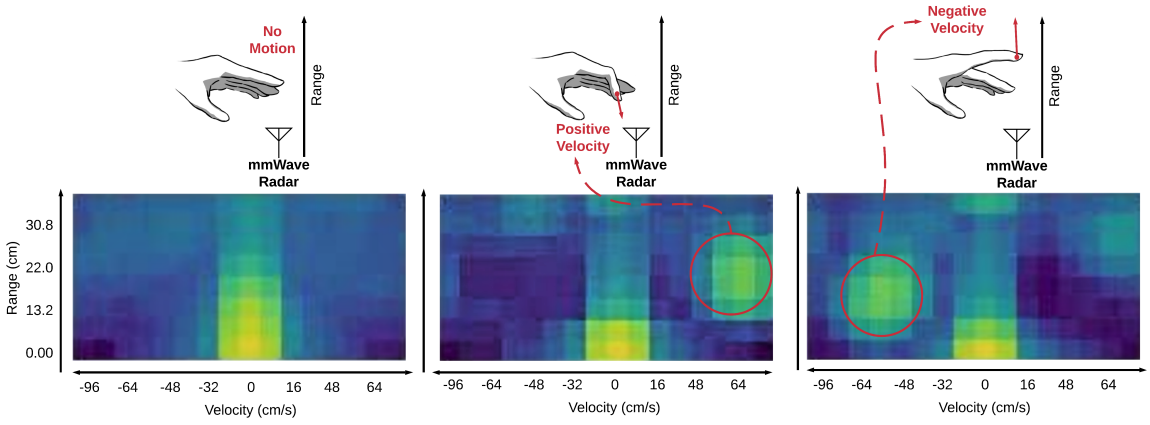

Fig. 4. The range-doppler profile captures the relative power of moving objects (such as fingers) in a particular *range*, or distance, from the radar moving at a certain radial *velocity* relative to the radar. In other words, it provides information about objects and their distance from, as well as motion towards or away, from the radar.

The number of rows correspond to the number of range bins. We used eight, making the radar able to see objects up $r_{max} = r_{res} \times numRangeBins = 35.2cm$. The number of columns corresponds to the number of the velocity bins, which we set to 16. It gives a maximum detectable velocity $v_{max} = \frac{v_{res} \times numVelocity}{2} = 1.28m/s$ (divided by two for positive and negative velocities relative to the radar). These parameters define the gesture performing range for the system. The number of velocity bins are constrained by the frame time, the data

processing time, and the transmitting speed of the serial interface. Considering the hardware processing and transmitting time, we found the best maximum detectable velocity to be $1.28m/s$ and with a FPS of 30 seconds. In our initial testing, we found that on average, the average speed of finger movement during an IndexPen gesture is $0.64m/s$, and is less than the maximum speed.

4.1.3 Range-Azimuth-Elevation Profile. Range-azimuth profile (Figure 5) is a dynamic feature set given by the radar. Obtained at the end of the signal processing pipeline, this profile represents angle of arrival (AoA) for the detected objects. Both the azimuth and the elevation combine into a matrix where the rows are the angle (azimuth and elevation) between the objects and the radar. The columns are the same as in the range-doppler profile; it shows how far away the objects are from the radar. The azimuth-elevation axis is non-linear as the angle resolution degrades as the objects moving away from the central axis of the sensor (Equation 6).

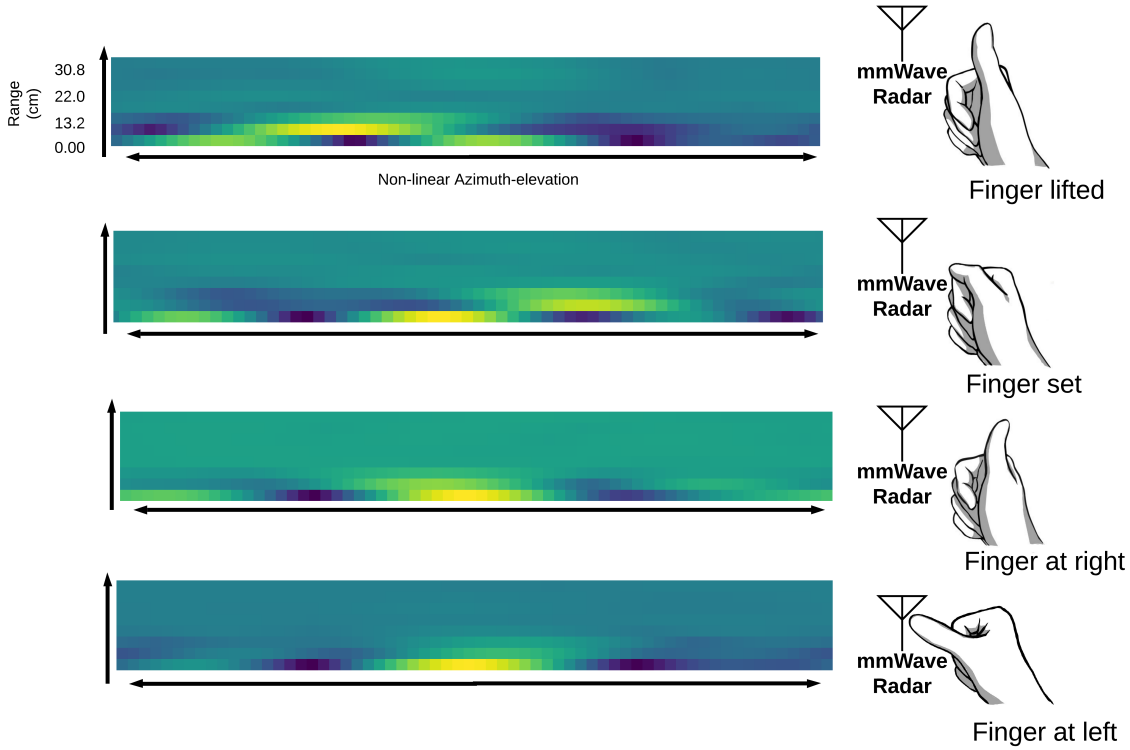

Fig. 5. The range-azimuth-elevation profile provides additional information that enables better micro-gesture detection, beyond the range-Doppler profile. Unlike the range-Doppler, which just provides velocity to or from the radar, the range-azimuth-elevation profile captures angle of arrival (direction) by giving the relative power for objects at a specific distance from the radar at a certain azimuth and elevation angle relative to the radar. Note that the images shown here are a concatenation of the range-azimuth and range-elevation with the horizontal axis representing both the azimuth and elevation.

4.1.4 Gesture Specifications. In the *IndexPen* system, we collect frames from the mmWave sensor, with each frame consisting of an 8×16 range-doppler (RD) profile and an 8×64 range-azimuth-elevation (RAE) profile.

Each frame lasts 33ms, giving 30 FPS. The sequenced frames are then used as the deep learning features in the next step of the pipeline to give the final *IndexPen* classification.

4.2 Clutter Removal Algorithm

By observing the raw RD and RAE profile, we find that the static noise from the direct path [35], as shown in Figure 4 and Figure 5, has a higher value compared to the motion's response. Therefore, we argue that the model can over-fit static noise which does not represent any feature related to the written gestures.

To demonstrate our hypothesis, we apply the pixel-wise clutter removal algorithm to each trail before extracting each sample using the event maker. Clutter removal is a widely used algorithm for static noise removal because of its low computation cost and simple implementation in a real-time system. Prior work [27] used this approach to remove static noise from the direct path between transceiver and receiver on ultra-wideband radar, significantly improving gesture classification accuracy.

The clutter removal algorithm is defined in Equations 4 and 5:

$$c_t = \begin{cases} ft & t = 0 \\ \alpha * c_{t-1} + (1-\alpha) * c_t & otherwise \end{cases} \tag{4}$$

$$y_t = f_t - c_t \tag{5}$$

f is the raw profiles from radar, c is the clutter frame, and t is the current timestamp. α within the range [0,1] defines the signal to clutter ratio that suppresses the new appeared static noise to α^t. The clutter at time t is defined as Equation 4. The clutter frame at current time t is produced from the residual clutter frame at $t-1$ and the data matrix at t. Then we subtract the clutter matrix by the raw data frame to get the clutter-free matrix y_t. In our case, we chose α equal to 0.8 which removes most of the static noise fast while sensitive to the motion. Figure 6 is the visualization of time series RD (top left and top right) and RAE (bottom left and bottom right) profiles before and after the clutter removal from time $t-4$ to t. We present the effectiveness of clutter removal for training process in section 5.1.3.

4.3 IndexPen Neural Networks

With the sensor-signal processing pipeline explained above, the physical information of the objects detected by the radar is represented in range-doppler, and range-azimuth-elevation profiles. As the data streams in, the sequence of the two profiles form a dynamic profile that is characterized by the gesture performed.

4.3.1 Neural Nets Architecture. In this section, we cover the multiple-input deep learning model that we constructed. The model extracts high-level features from the two types of profiles aforementioned and output the *IndexPen* classification.

We design the network structure with the following considerations. It has been shown that Convolutional Neural Networks (CNN) are well-suited for distilling two-dimensional features such as images. CNN captures the spatial feature of the image-like profiles and is able to extract non-linear, and high-level features such as whole finger motion, the relative location of specific hand parts, and so on.

Meanwhile, the input contains two profiles, and the model should first merge them. In solving similar problems, past studies have added a channel dimension and put the multiple images as different channels [62]. For this system, it is also important to note that the relevant features for gesture detection in range-doppler and range-azimuth-elevation can differ greatly as the former reflect the radical velocity, and the range-azimuth-elevation is the spatial information. This fact led us to use two different CNNs for the two profiles, and they do not depend on each other.

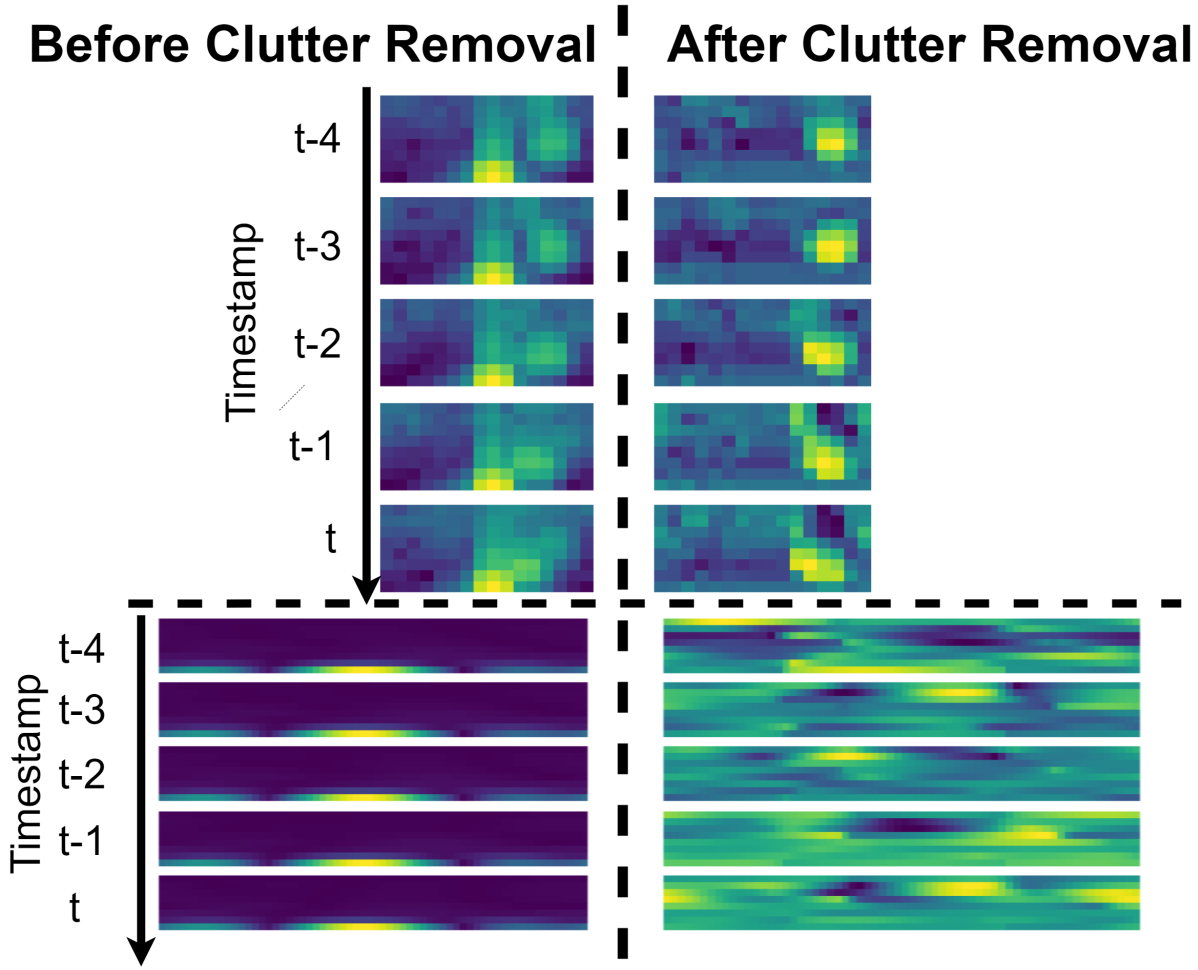

Fig. 6. This Figure shows the RD and RAE profiles before and after clutter removal for five continuous frames. Top left: Range-Doppler profile before **clutter removal**. Top Right: Range-Doppler profile after **clutter removal**. Bottom Left: Range-azimuth-elevation profile before **clutter removal**. Bottom Right: range-azimuth-elevation profile after **clutter removal**. The user was moving his hand toward the radar. The **clutter removal** algorithm applies exponential decay to the data points that have not changed between the current time point and the previous as shown in Equations 4 and 5. Similar to the weighted moving average [21], it assigns more weight to more recent movement. We can see the static noise in the middle of the profiles in the raw RD and RAE profiles shown in the left column whereas their clutter-removed counterparts in the right column is more descriptive of the current movement.

The extracted feature maps from the two profiles are flattened and concatenated before being sent to the time distributed layers. As the features are coming in as sequences, time has to be taken into account. That is, the model needs to look at data in a time interval to make predictions. As traditional neural networks suffer from the exploding/vanishing gradient problem when dealing with sequenced data, we use long-short-term-memory (LSTM) layers to solve the problem. LSTM has the capability of gating the information, giving the model another

level of flexibility to decide what temporal features are important for the gesture detection. In this work, the interval was set at 4 seconds as we find this is a typical time to perform one writing with *IndexPen*. Given that the frame rate of the hardware pipeline is 30 FPS from Section 4.1.2; the number of time steps that the LSTM layers takes is 120 ($30 frame per second \times 4 second$)

Indeed, prior work [61] showed that Convolutional Recurrent Neural Networks (CRNN) with LSTM cells perform well in resolving the range-doppler profile in gesture classification. Recent work extends this idea to that of the range-azimuth-elevation profiles as a second input for the network. Other work [68] demonstrated that the fusion of two profiles allows the feature extracted from angular information, improving the classification accuracy of gestures with similar range information but different moving directions. In addition, our work applies a clutter removal algorithm [27], a recursive denoising algorithm, on both range profiles, significantly reduced the interference from static noise and showing a remarkable improvement in the robustness of the system's ability to detect micro two-finger gestures.

After extracting the spatial and temporal features, the system takes to the standard fully connected (FC) layers. We make a copy of the extracted features and put them through an FC layer with sigmoid activation functions that output the *IndexPen* classification predictions.

With that, we proposed the following neural network model as shown in Figure 7.

As *IndexPen* is intended to be used in edge computing devices, to facilitate real-time interactions, the network architecture should be lightweight and low latency for the users to use in practice. With some preliminary experiments on the trade-off between accuracy and model complexity, we devised the network to contain two levels of convolutional operations, two LSTM layers, and two FC layers for yielding the tracking and classification results.

4.3.2 Optimizations to Improve Robustness. To combat over-fitting, we used the standard practice of randomly **dropping nodes** trained in the system by specifying dropout rates indicating the percentage of neuron units to mute in making a prediction with forward propagation. This helps reduce the output dependency on certain features [57].

Moreover, our preliminary experiment showed that even with the above methodologies that facilitate model robustness, the validation accuracy is far less than desirable when the input dataset is composed of samples from multiple users. Sections 4.1.2 and 4.1.3 detail the two types of features that the system uses for dynamic gesture classification. It is worth reiterating that the above two features have a high temporal resolution (i.e., range-Doppler profiles yields speed and range measurement every 33 ms), making them prone to noise and interference. In real-world settings, random motion in the environment generates background noise in the input features. Noise can come from sources such as the involuntary tremor of the gesture-performing digits, the rest of the body, or other people present in the scene. Such random motions are challenging to remove algorithmically because of the difficulty to distinguish the undesirable motions from the actual gesture.

At the same time, we adopt the commonly used method of windowed classification. We observed that the four seconds of a dataframe usually only contains classification-critical information that is less than half of the duration. Therefore, the model needs to account for the sparsity of features in the time window. Additionally, sparsity is characterized spatially as well as temporally. The radar's field of view, in order to capture the full image of the hand in the discussed mobile use cases, is set to be a semi-sphere with a radius of 35.2cm in front of the sensor device. As shown in Figure 4, the dynamic hand motion is only reflected in a comparatively small sub-image of the whole range-Doppler profile; the result would be best to not be considered in the identification of the gestures.

Regularization is the method applied in the *IndexPen* neural network architecture as a within-neural-nets method of denoising and dealing with the sparse feature. Regularization is the weight penalty added to the neural nets' layers that encourages the weights to become smaller; in other words, it weakens the connection between the

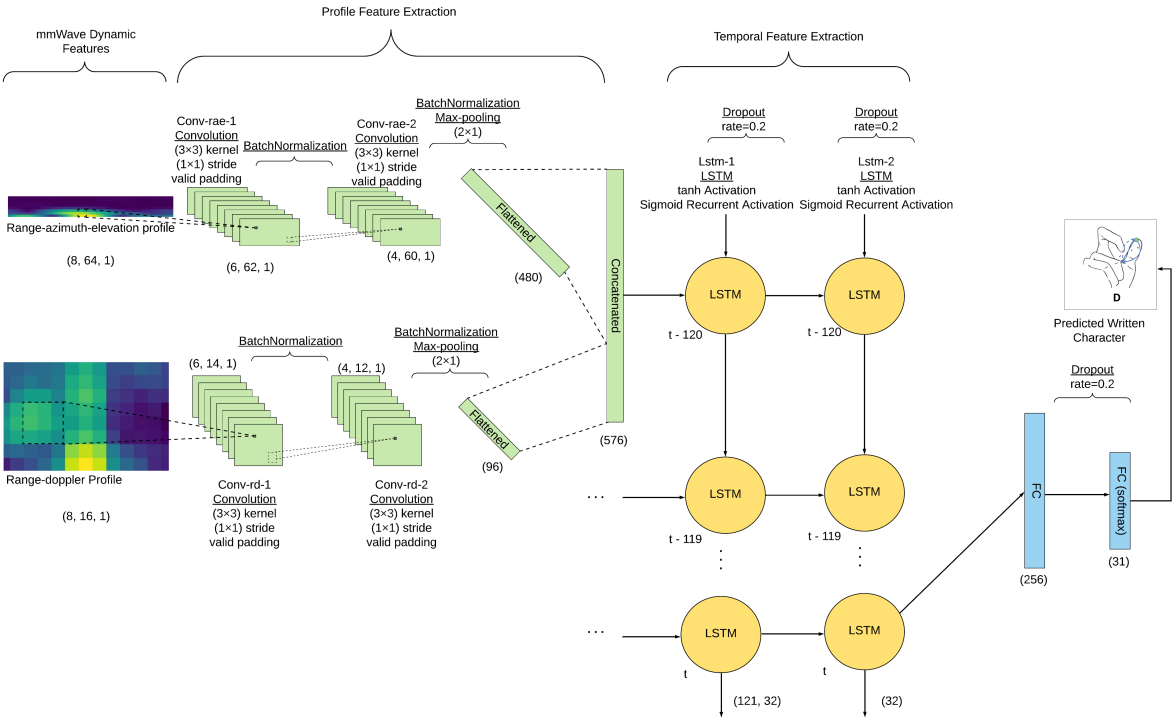

Fig. 7. IndexPen CRNN Architecture: the two inputs include the profiles for range-Doppler (RD) and range-azimuth-elevation (RAE) profiles. 2D convolution is then performed on each of the profile feature maps which is fed into LSTM layers. LSTM cells further extract the temporal information and send copies of their output to the *IndexPen* FC layers which give the final prediction of the gesture detected. The RAE features (the heatmap at the upper left) are added to the *IndexPen* model and compared with the range-Doppler-only model. That is to see the effect of adding range-azimuth-elevation has on the performance.

perceptions in general so that the neurons that actually matter (neurons connected with the classification-critical information) can stand out. Namely, *kernel regularizer* and *recurrent regularizer* are applied to the convolutional and recurrent layers, respectively. *Bias regularization* is also applied to the above two types of layers to encourage the model to reason more from the original input than from a specialized bias term, which is prone to get over-fitted in this case. Lastly, we apply an *activity regularizer* to limit the output strength to facilitate other regularizers, and in part also to reduce over-fitting. We show in the first part of the user studies that the introduction of regularization terms greatly improves the robustness of the model. The regularization hyper-parameters are not explicitly discussed in this paper as they are specially tuned for the sensor and data preprocessing pipeline we built. Interested readers may find the source code in the provided GitHub repository.

4.4 Real-time Gesture Detection

The sigmoid activation function at the output layer yields a vector that indicates, in the past 120 timesteps (i.e., 4 seconds), the probability that each of the gestures has been written. The size of the probability vector is thirty given the thirty classes for classification. The prediction is made on a per-frame basis to give the size-30 vector at every time step.

Algorithm 1: *IndexPen* detection algorithm with debouncer to stabilize input sequence

frame - the current radar frame;
frameBuffer-the queue that holds the frames of the most recent classification window;
debouncer - a list of 30 integers, one for each *IndexPen* class;
debouncerFrameThreshold - Constant, the number of frames needed for the debouncer to register a detect;
debouncerProbThreshold - constant, the minimal probability needed for the debouncer to consider a gesture is being performed;
relaxPeriod - constant, the number of frames to ignore after a detection. This is to reduce the error when writing consecutively same characters.
relaxCounter - ticks during the relax period and raise flag when it counts up to the relaxPeriod
Result: Per-frame detection of gestures *e*

```
while frame do
    frameBuffer.push(frame) ;
    if relaxCounter == relaxPeriod then
        yPred = model.predict(frameBuffer) ;
        breakIndices = argwhere(yPred>=debouncerProbThreshold) // in single-frame, letter probability is
          above threshold ;
        debouncer[breakIndices] += 1 ;
        lowerIndices = argwhere(yPred<debouncerProbThreshold) // in single-frame, letter probability is
          below threshold ;
        debouncer[argwhere(debouncer>0)  lowerIndices] -= 1 // decrease the debouncer counter for the
          letter whose probability is lower than the threshold
        detects=argwhere(debouncer >= debouncerFrameThreshold) // over period of time, letter
          probability remains above threshold ;
        if any(detects) then
            // if a gesture registers an input after debouncing ;
            stdin(classes[detects]) // outputs detected keystrokes ;
            debouncer = [0] * 30 // reset debouncer ;
            relaxCounter = 0 // enters relax period following detection ;
        end
    end
    else
        relaxCounter += 1 ;
    end
end
```

As the classifier deals with streams of data, it is also important to see how the predicted probability evolves as a gesture is being performed.

Figure 8 gives the evolving predicated probability as the radar frames are being fed into the network while the user is performing the gesture. It is evident that the probabilities for all the gestures show a sudden surge upward at the second after the gesture onset. Moreover, the probabilities all tend to converge towards the end of the third second. We note that it is necessary to set thresholds value for both the probability and number of frames with high probability for determining if an input has actually been made.

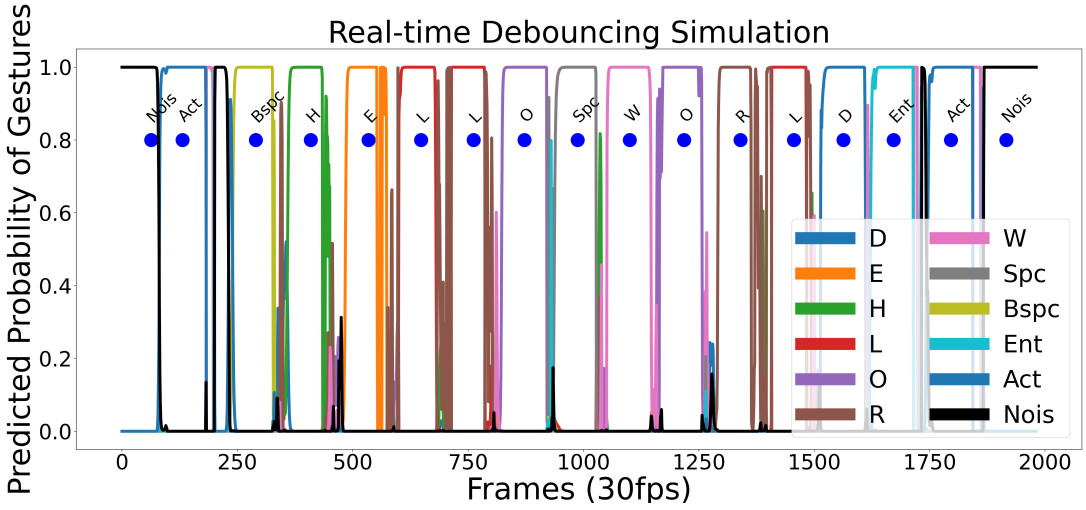

Fig. 8. The temporal probability evolution of Participant P 1-1 (see Section 5) input of sequence "Activation-BackSpace-Hello-Space- World-Enter-Deactivation". The predict probability of each class shows a arbitrary jittering behavior. The proposed real-time gesture detection algorithm can successfully output keyboard-like character inputs (indicated in blue).

As seen in Figure 8, the probability of each gesture given by the neural network changes rapidly, which is partially due to the random motion or tremor in the hand. Further, similar to handwritten characters, some gestures in the *IndexPen* alphabet have a similar signature (e.g., P and D, the only difference being the coverage of the curved stroke). To enable high-precision character input, we apply a simple yet effective debouncing algorithm.

We implement a simple software debouncer that keeps a size-30 integer vector as the counter for each class and is called whenever a prediction is available. For a switch that has a state either on or off, instead of using the observed state, which can lead to unstable behaviors, a debouncer algorithm is usually implemented with a counter and sliding window: only when the observed value of the switch is in a certain state for the pre-defined duration, will the digital system register a button press [37]. For example, debouncer approaches are commonly used for electrical buttons (e.g. a key on a keyboard). Buttons implemented with contact switches have the tendency to impulse signals in the time domain making it unclear for the digital system to determine if the button is pressed. Our algorithm takes in thirty probabilities values given by the neural network at each frame; it checks if any of the probability passes a pre-defined threshold; for those that do, it increments the frame counters for those classes. When the frame counter of any gesture passes a frame threshold, *IndexPen* fires to the input interface that this character associated with the counter is written. This would be equivalent to pressing this key on a physical keyboard. After a key is fired, we reset the counter and continue the process.

The pseudo-code for the detection pipeline, with debouncing, is in Algorithm 1, which runs whenever a new radar frame becomes available. For similar gestures, the probability of an incorrect character may reach the *debouncerProbThreshold*, but the debouncer prevents the incorrect classification by requiring the probability to remain above the threshold for a number of frames (*debouncerFrameThreshold*). For example, we set the constant *debouncerProbThreshold* to 0.7 and the constant *debouncerFrameThreshold* to 50 frames. This means that the algorithm will narrow down the correct gesture when the probability remains above 70% for 50 frames, which is approximately 1.7 seconds. The total time may vary, depending on the values of these constants as well as the gesture and the user's performance.

5 STAGE I: TOWARD A PRELIMINARY PRE-TRAINED GESTURE DETECTION MODEL

To explore the feasibility and potential of *IndexPen*, we present a two-part study that builds a foundation for future work on *IndexPen*. First, we show that we can build an accurate model of 30 gestures on a dataset collected over 10 days, and explore variants of the modeling algorithm to determine the most robust model. We then explore the extent to which a model can adapt to a new user and the amount of samples needed to do so. These studies were approved by the WPI Institutional Review Board.

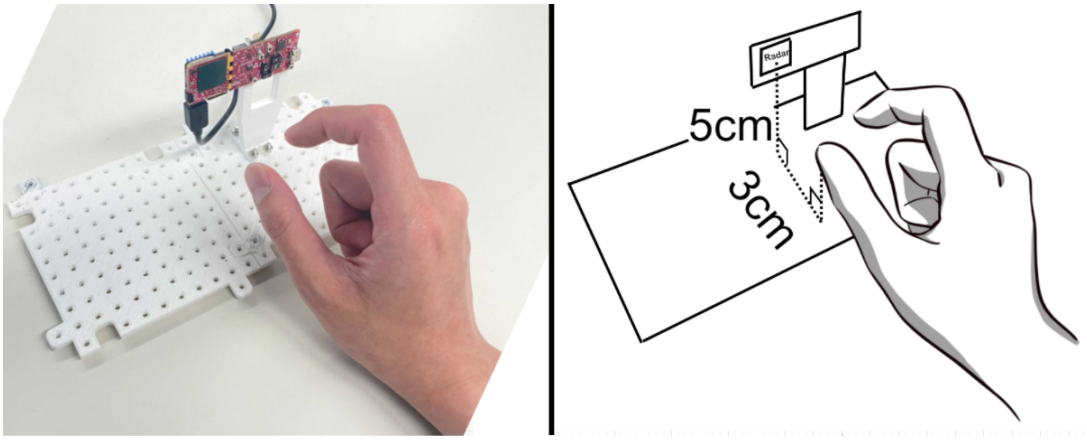

Fig. 9. This Figure shows the experiment setup for data collection and the user studies (Sections 5 and 6). The users were instructed to put their hand at the same height (5cm) as the radar and keeps a distance of 3 cm away.

5.1 Part 1: Validating the IndexPen Classification Model on 30 Gestures

The goal in this part of the study was to collect a large dataset enabling the development and evaluation of a pre-trained gesture detection model for *IndexPen*.

5.1.1 Participants. We recruited five participants (*5 males*), with an average age of 24, labeled as *P1-1 to P1-5* (e.g. P1-2 refers to participant 2 in user study Stage I). *P1-1* and *P1-3* were familiar with *IndexPen* gesture before the experiment and the other three participants had never been exposed to *IndexPen*. All participants reviewed and signed an informed consent form prior to participating in the study. While the number of participants is relatively small, the data was collected over 10-12 hours from each participant, enabling for a large dataset from each person.

5.1.2 Data Collection. During the user study, the user was instructed to put the tip of their index finger and thumb at approximately three centimeters from the radar, and at the same height as the center of the radar. This way we maximize the angular resolution of the hand motion because it is at the center axis of the radar and as close as position without touching the device. The setup is depicted in Figure 9. We also note that the placement of the hand is not strictly enforced and is subject to individual discretion. We assume that minor displacement in gesture location is advantageous in helping training a more robust recognition model. In general, a displacement within 1-2 centimeters is tolerated.

The data collection for each participant happened over 11 different sessions, each lasting about one hour. The first day was the training for writing gestures. For the next 10 sessions, each day contained 10 trials in which the participant was instructed to write each of the 30 characters along with a *noise* sample twice with randomized

Table 1. Stage 1 *IndexPen* overall validation accuracy across all character gestures and all users. The best accuracy without clutter removal is only 3.4% lower than the model with clutter removal. However, referring to Figure 10, the training process is much more stable with clutter removal.

	Validation Accuracy
IndexPen Without Clutter Removal	0.924
IndexPen With Clutter Removal	0.959

order during a fixed interval. We define a sample as the data stream produced by the radar in the 4 second window during which a specific character is written. Therefore, we collected 620 labeled samples from each participant every day, and 20 samples from each of the 30 characters as well as noise samples.

In total, over the 10 sessions, we collected 6200 samples from each participant, with 200 samples per character/class/subject. Each sample consists of 120 radar frames (4 $sec\ per\ sample \times \frac{1}{30ms}\ frames\ per\ second$) made of the range-doppler and range-azimuth-elevation profiles. Overall, this part of the study collected 3.72 million frames (6200 $samplesper\ subject \times 120\ frames\ per\ sample \times 5subjects$) for both aforementioned radar profiles.

5.1.3 Classification Architecture and Variants. We shuffle and stratify split the 6200 samples for each character from a different participant from different days. With this large dataset, we set the training and validation ratio equal to 9:1, giving the model more generalizability while having 100 $(20*5)$ samples from each class in validation set. In training the *IndexPen* network, we used the adaptive moment estimation (Adam) [5] optimizer with the initial learning rate at $1e-3$ and decay of $1e-5$. According to the suggestion from [56], we set the batch size to 128. With early stopping call-back with the patience of 250 epochs, the network converged at 415 epochs. We explored the difference in classification accuracy with and without clutter removal.

5.1.4 Results and Discussion. The overall validation accuracy across all characters with and without clutter removal preprocessing is shown in Table 1 and the training history is shown in Figure 10. The best-performing model for classifying the 31 classes is 95.89%, and the accuracy for each individual class is higher than 91%. This high classification accuracy demonstrated the feasibility of our designed system as a text input system.

While there is not a large difference with and without the clutter removal, the large fluctuation for validation accuracy and loss shows that the model can easily over-fit to unrelated components with small misalignment. As shown in Figure 10, keeping all the settings the same, the loss and accuracy gain much higher stability after adding the clutter removal algorithm as we discussed in Section 4.2.

The model described above was trained by 180 samples per class from each participant, and we observed the writing style varies from user to user during the data collection. For example, participant 1-1 created the last stroke for 'O' downward while participant 1-2 created it with a forward movement. Additionally, the 'D' for participants who have a smaller hand could be very similar to the 'P' from participants with larger hand size. Thus, it is important to handle individual differences.

5.2 Part 2: Exploring User Adaptability with Transfer Learning

It took more than 10 hours to record 180 samples/class for the model above, which may be an unrealistic requirement in a commercial product. In this section, we use the same dataset as in the previous section. However, we apply the leave-one-out method to build a model on four users and see how well it performs on a new fifth user, and also how much data from the new user is required for a robust model. We aim to reduce the number of training samples required while keeping the classification accuracy at the same level. In addition, we were interested in exploring the best strategy for transfer learning. Some prior work retrains the last dense layer

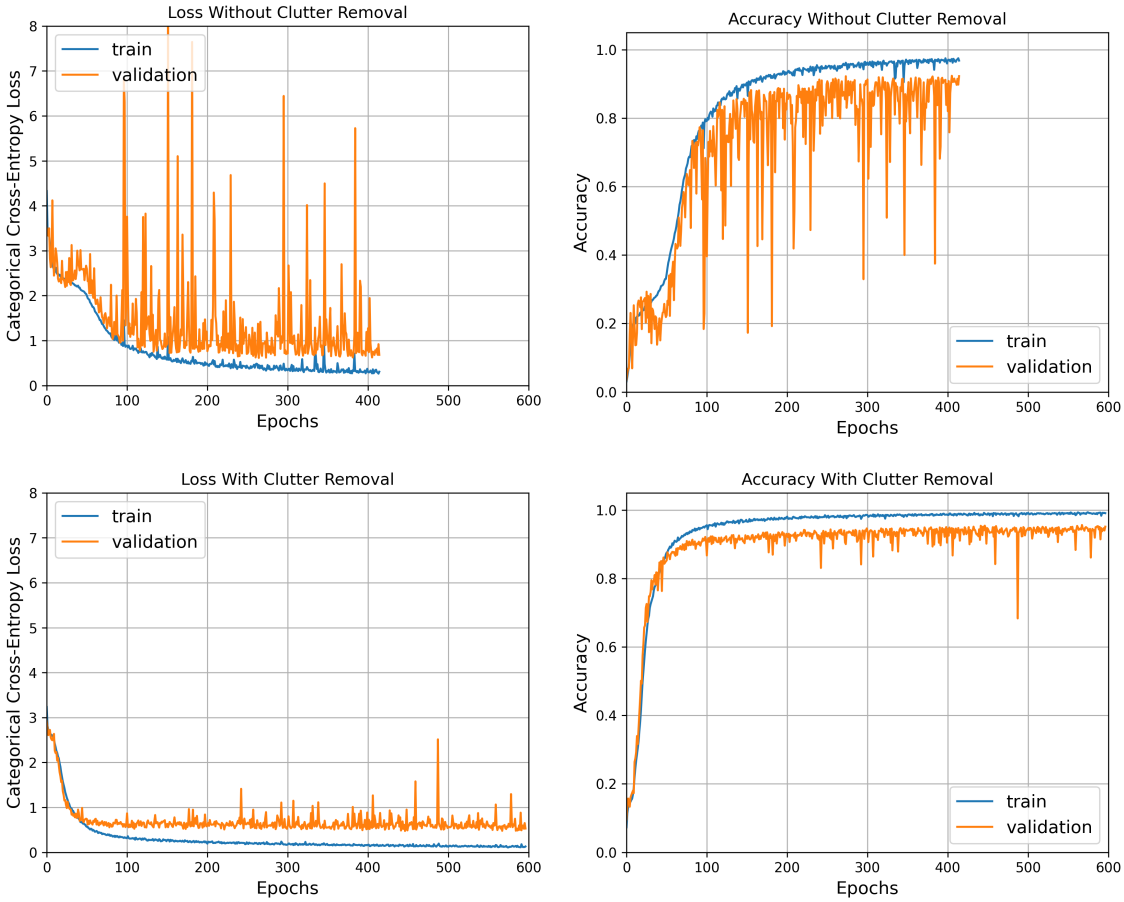

Fig. 10. Training history of the complete *IndexPen* neural network architecture. The top row is the training history without clutter-removal algorithm and the bottom two charts is the training history with clutter removal. Although the accuracy for the model with clutter removal is only 3.4% higher than the one without clutter removal, comparing two variances, removing the static noise increases the stability of the model as shown in the reduced variance.

only, which works well when there is high generalizability of the model. Other work has retrained all layers. We explore both approaches below.

5.2.1 Methodology. We were interested in finding out how much calibration data we need for a new user with an existing model and what method to use for the transfer learning. We left out one participant and trained the model on the other four users' data, keeping all the settings the same as in the last section.

To study the amount of required calibration data, we randomly sample 20 samples from each class from the left out user and gradually increase the number of samples for transfer learning and use all the rest samples (180 samples/class) for testing. Starting without feed-in samples, we increase the feed-in ratio by 10 percent (2 samples/class) each step. In order to regularize the model with insufficient data and combat over-fitting, we

decrease the learning rate to $4e-5$ and decay to $2e-5$. In addition, to reduce the variance from the random split, we repeat this entire process 10 times and select different samples for transfer learning without repetition.

As discussed above, we tested two transfer learning strategies: 1) *All Layers Trainable* where we unfreeze all network layers during transfer learning, and 2) *Last Layers Trainable* where we retrain the last dense layer only. Both strategies are widely used [47, 64, 65] for different tasks, and [10] compared results from the two strategies for text recognition with *IndexPen*.

5.2.2 Results and Discussion. The average validation accuracy for the five pre-trained models is 92.48%. We compared the result from two different transfer learning approaches (*All Layers Trainable* and *Last Layers Trainable*) and the result is shown in Figure 11. The second method shows a much higher learning curve with the same number of samples for transfer learning. In general, with 20 training samples, *All Layers Trainable* model outperforms the *Last Layers Trainable* model by 16.75%. Therefore, we argue that the feature space in the pre-trained model is not general enough to interpret a new user's writing style. The reason is that the writing style for the same gesture could be very different between users. Training only the last layer is a constraint on how much the model can adapt to a new user. Most spatial and temporal features are abstracted by the convolutional and regression layers, which are not trainable in *Last Layers Trainable* settings.

Additionally, when all the layers are trainable during adaption training, a pre-train model is more susceptible to being biased by the variance in the training data given. The vulnerability to between-user variance is shown by that the error bar in *All Layers Trainable* is greater than *Last Layers Trainable* when feeding fewer samples. Nevertheless, the accuracy converges towards the mean (has smaller error bar) when the model sees more training samples for the same user. The mean accuracy of *All Layers Trainable* is larger than *Last Layers Trainable* because the entire model was able to condition on the distribution of new users' gestures.

The sharp accuracy drops we see when we validate the model on a new user motivate the need for a user-specific calibration process. After applying transfer learning and increasing the number of samples for training to help the model adapt to a new user's writing style, the classification model achieves a relatively high accuracy. The calibration using 20 samples/class increase the average accuracy across 5 participants from 65.58% to 87.55%.

The notch at 0.1 (2 samples/class) feed-in ratio is caused by over-fitting, and we discuss the possible solution in Section 7.

Insufficient data is a common problem in many domains. Data collection often is time consuming and expensive making the training process extremely difficult and biased. For example, building a brain model using an MRI image [11] involves an expensive experiment setup and a high variance between participants, which increases the number of required samples from the larger population. Transfer learning, as a solution to this problem, mitigates the demand that the distribution of training data must be identical to the testing data and validation data. Many recent works have demonstrated this hypothesis [44]. For example, [54] has been used as a transfer learning baseline network [58] in image classification, object detection, segmentation, as well as gesture motion detection [2], because the feature distribution in the pre-trained layers is similar to the corresponding problem.

5.3 Stage I Summary

In the first stage, we built the preliminary *IndexPen* model and explored some of the considerations. This model is the foundation for Stage II described below.

6 STAGE II: TOWARD REAL-WORLD USE OF *INDEXPEN*

In the second stage, we conducted a multi-part study to bring IndexPen closer to real-world usage. Instead of collecting data on one character at a time, we explored a real-world writing task, using sentences. We explored transfer learning across a larger group of users. In addition, we explored the learning process of both the algorithm

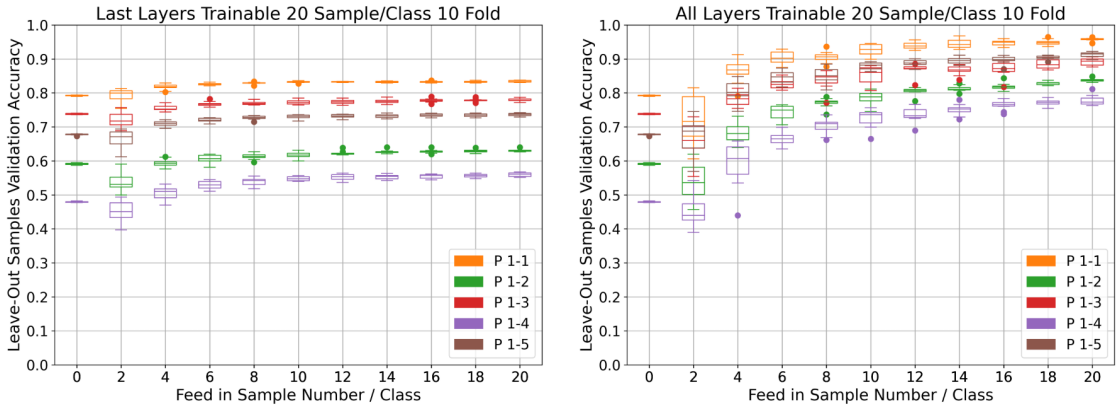

Fig. 11. Leave-one-out transfer learning across participant 1-1 to participant 1-5. The x axis represents the feed-in ratio of 20 samples/class and y axis shows the validation accuracy from all the rest samples from the leave-out participant. Each box represent 10-fold transfer learning cross validation. *Left*: all layers trainable model. *Right*: only last layer trainable. Starting with average accuracy equal to 65.58, all the layers trainable model achieves 87.55% average accuracy at with 20 feed in samples across five participants while the last layer trainable model is 70.8% with the same condition.

and the participants over five sessions. Finally, we elicited suggestions and perceptions about the potential of *IndexPen* for human-computer interaction.

Unlike recognizing static images such as signatures, gesture detection carries the additional complexity related to the dynamics with which a subject moves his or her hand, or the mode of motion/gesturing. It is known that this temporality of the features carries significant individuality [29] and raises the question: can a model obtained from a group of users be adapted to new user groups? In Stage I, we showed our system can successfully classify 30 distinct gestures with high accuracy, and the transfer learning is capable to capture the writing variance between individuals.

In this second stage, we present an investigation of the cross-user adaptability of the *IndexPen* gesture system. We look at how our model performs on a new user's data and whether we can improve the initial model by calibrating to the new user. Moreover, we evaluate the user experience with a three-part survey: pre-experiment screening, post-session interview, and real-world scenario survey. As in Stage I, this study was approved by the WPI Institutional Review Board.

6.1 Participants

We recruited 16 participants (8 male, 8 female), with average age 22. None of the participants in this group had been exposed to *IndexPen* before. Among them, one participant (P2-11) was left-handed where the pre-trained model was on right-hand data only (Section 3.1) collected on right-handed people. This participant still learned the gesture with his right hand. We label all the participants in this study as P2 − *n* to distinguish the participant from previous study. At the beginning of the study, all participants reviewed and signed an informed consent form.

6.2 Procedure

During the experiment, participants were instructed to write sentences with the *IndexPen* gesture and receive real-time feedback of the text they 'typed' in a text box on the computer screen. Participants were asked to come

to five sessions in total, on five different days. Each time they completed a session, the pre-trained model was retrained to adapt to this user. The transferred model was used to make real-time inference when the user came in the next time.

6.3 Experiment and Tasks

We developed three applications (Figures 20, 19, 21) for data collection, visualization, and real-time inference. The detailed software implementation is in Appendix A.3.

This experiment followed the procedure shown in Figure 12. We prepared 50 pangrams - sentences that contain all of the letters of the English alphabet. In each session, participants were instructed to write 10 different pangrams in a text box. Each sentence contained 37 characters (gestures) on average. Since the sentences did not contain some of *IndexPen's* special characters including *Backspace*, *Enter*, *Activation*, and *Noise*, we changed each sentence sample to:

$$Noise - Activation - Backspace - < APangramSentence > -Enter - Activation - Noise$$

We added two *Activation* at the head and the tail of the sentences to simulate the user activating/deactivating *IndexPen* at the beginning and the end of typing a sequence. We also put one random character in the input box, so the user needed to perform some *Backspace* gestures to be cleared for writing the requested sentence. Similar to the first session, the participant followed the instructions on the user interface, writing each gesture in a four-second interval. Using a fixed interval simplified the labeling data for a specific gesture. The real-time inference algorithm invoked a keyboard strike for each detected gesture, providing real-time feedback to participants. It enabled the participants to see what the system inferred that they typed (correct or incorrect). The participants were instructed to keep writing even when they made a mistake (i.e., writing F whereas they should write T or forgetting how to write the gesture).

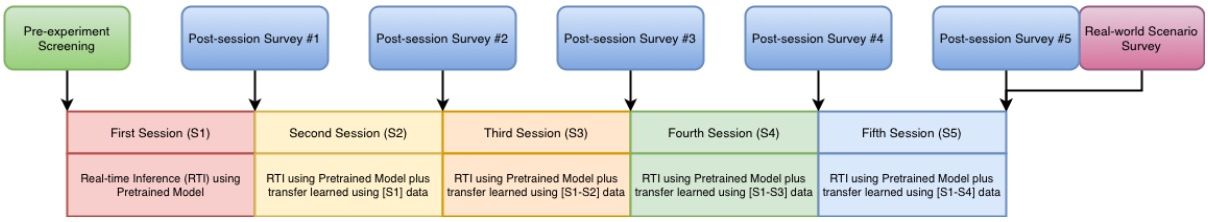

Fig. 12. The flowchart of the sessions in user study Stage II (Section 6). We started every participant with a pre-experiment screening. Then the user went through five sessions. The real-time gesture detection model used in the first session was the pre-trained model from Stage I (Section 5). After the first session, the model was transfer learned on data from the past sessions of the same user (i.e., the model used in session 3 would be the pre-trained model retrained on this user's data collected through session 1-2). At the end of each session, the participant was given a post-session survey. After the last session, the participants answered a *real-world scenario survey* about how they thought *IndexPen* could be incorporated into existing hardware applications.

At the end of each session, we gave the user a post-session survey for them to reflect on their experience with *IndexPen* so far. We conducted a semi-structured interview where we asked which gestures were the most challenging and how much easier it was compared to the previous session.

6.4 Performance Measures

To explore real-world use, we investigate the mistakes user made in real-time while interacting with the system, the model accuracy in various settings, and how user experience changed as they learned more about *IndexPen* and the model adapted to their gesture style. Namely, we utilize the following measures:

(1) **Sentence Correctness**: To explore the correctness of the sentence we measure string similarity during real-time inference (RTI). To do this, we use the *Levenshtein distance* [19] between the requested sentence and the sentence that the participant actually typed through *IndexPen* (the character sequence that the model decoded).
(2) **Incorrect Gestures**: In all sessions, a researcher observed the participants and identified any incorrect gestures, noting only those that are completely incorrect. Often the participant also realized that the gesture was not correctly executed. These were noted both for exploring user performance and learning curve and also for ensuring that the classification model was not trained on incorrect data labels.
(3) **User Model Performance with Transfer Learning**: After each session, when new data became available for a user, we re-trained the pre-trained model from User Study I on a 80-20 train-test split. The user calibration was evaluated by the *transfer-learned model's F1 score* showing the accuracy of the re-trained model on the test data for this user. We use the F1 score to evaluate the prediction accuracy because the sentences that the participant wrote have a different number of samples for each gesture. We further explore the variation in F1 score across different gestures.
(4) **User experience** we make a qualitative evaluation of the system based on the participants' feedback before and after the sessions. We examine each participant's experience coupled with the previous two metrics.

6.5 Results and Discussion

Here we discuss the results of the second user study about the IndexPen's user learnability and model adaptability in more realistic settings.

6.5.1 Sentence Correctness. Each box in Figure 13 consists of ten similarity scores using the Levenshtein distance from the ten different pangram sentences in that session. The triangle highlights the median value in the box chart. We observe that that participant makes fewer mistakes after the second session and the sentences continue to become increasingly correctly entered.

6.5.2 Incorrect Gestures: Incorrect gestures happened most frequently during the first session. Participants on average made mistakes (writing errors) on 1.58% of the 37-letter long sentences; the percentage incorrect is reduced to 0.31% for the sentences on the last session. It shows that the users gradually learned to create a natural map between the gestures and the characters, enabling more fluent interaction.

6.5.3 User Model Performance with Transfer Learning. We present the average F1 score across characters for each participant and the same metric across all participants for each gesture. We show how the accuracy evolved as participants came in for each of the five sessions in Figure 16. We compare the results using a static pre-trained model and that with transfer learning on the user-specific training data from each session.

As shown in Figure 14, the average F1 score for both the transfer model and the original (static) pre-trained model are equal to 51.8% on the first session, because no user-specific data is available for transfer learning/calibration in the first session. With transfer learning, the model's accuracy increased to 88.3%(δ36.5%) by the last session. It is interesting to note that the F1 score of the pre-trained model (without any re-training) also went up substantially to 65.7%(δ13.9%). One hypothesis on the improvement of the F1 scores is that the users also learned to adapt to the system over the five sessions, as they were able to see the detection result in real-time. While the model was calibrating to the user with transfer learning, the users were also adjusting their gestures to get better results from the system.

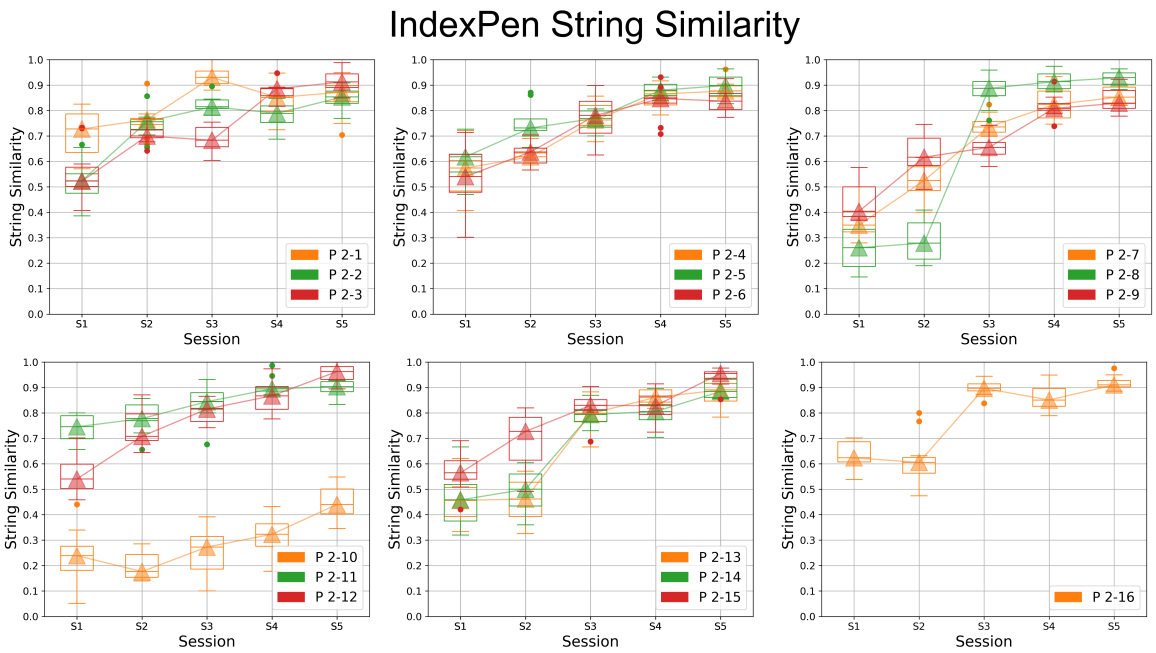

Fig. 13. The string similarity from the real-time inference in the user study. Each box consists of ten similarity scores from ten different pangram sentences.

6.5.4 Model Performance Variance between Different Gestures: Figure 16 presents the F1 curve for each individual gesture across all users. Figure 15 shows the detailed accuracy for each letter and each participant in the last session. (The detailed F1 scores for other sessions are in Appendix A.4)

It is interesting to note the gestures $U(\delta F1 = 54.0\%)$, $Backspace(\delta F1 = 51.0\%)$, $V(\delta F1 = 51.0\%)$ in the first sessions. These gestures had the highest accuracy increases from session to session, indicating that there was high variability in how each participant wrote these gestures. Towards the final session, the model was well calibrated with transfer learning to cope with those individual differences in the mode of writing. W had the highest accuracy (95.5%). It is one of the longer gestures involving the greatest number of strokes creating long temporal features that make its classification easier. $Activation$ started with high accuracy even in the first session and ends high, increasing from 83.9% to 95.8%. This gesture includes a double tab which requires the least manual dexterity with minimum variance between individuals. We noticed that the model incorrectly predicts all the F gestures as a T on the last session for *P2-2* (There was a 10-day gap between session 4 and session 5). After reviewing the video recording, we found that instead of keeping the thumb relatively stationary as in previous sessions, *P2-2* kept the index finger static and wiped the thumb downward for the first stroke in F. The motion of the thumb with respect to the index finger is correct; however, this motion with respect to radar is reversed. Therefore, the F and T are very similar on the range profiles. We discovered the same problem for *BackSpace* and *Space*. This variance also indicated the writing style changed over time [6].

6.5.5 User Experience and Performance. In this section, we present the analysis of the user experience with analysis combining the post-session survey and the quantitative evaluation. We also cover the participants' reflection on the most challenging gestures and their learning experience in the survey. In the final post-session interview, most users noted a high improvement compared with the previous sessions, and the sentences becomes

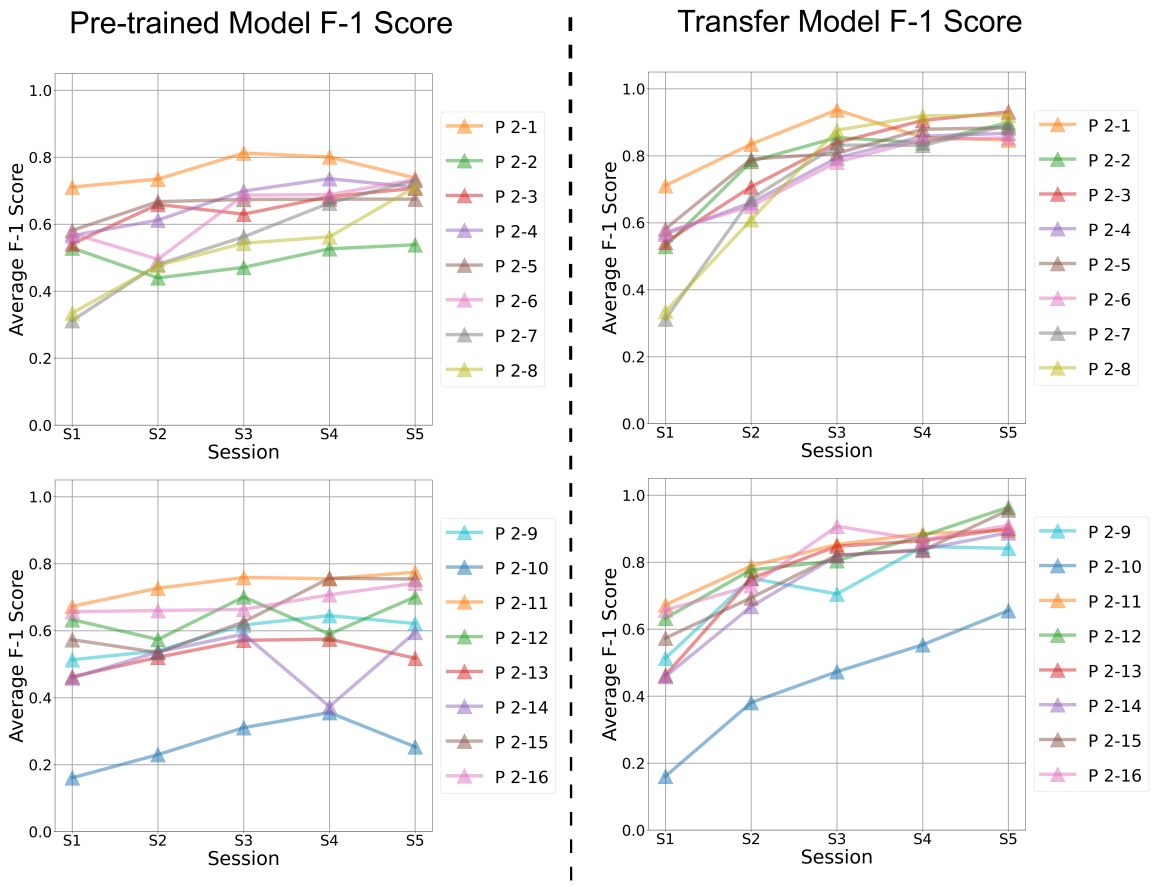

Fig. 14. The average F1 score from the raw sample between event markers across all the classes and participants over five sessions. Left: average F1 score using the pre-trained model. Right: average F1 score from the transfer-learning model. We separate the 16 participants into two different plots for better readability. The pre-trained model accuracy and the transfer learning model accuracy start at the same level. However, the transfer learning resulted in significantly higher accuracy towards the last session.

readable after the third session. Among the participants, participant P2-8, P2-7, P2-13 showed the highest adaptation rate throughout the sessions. The average accuracy increments for those three participants are ($\delta F1 = 58.6\%$), ($\delta F1 = 57.9\%$), and ($\delta F1 = 43.9\%$) respectively.

Participant P2-15 made the following observation during the post-session survey at the final session, noting how much improvement they had experienced since the first session.

> "*It's frightening how accurate it is considering the measly five sessions of the experiment. The gesture has also become more comfortable and a lot more accurate. When a wrong letter appears, it is very easy to understand and analyze where the mistake might have come from. There is also a pattern of the mistake appearing mostly in the middle of the test as the hand becomes more worn out. This seems to be improved whenever a break is taken.*"

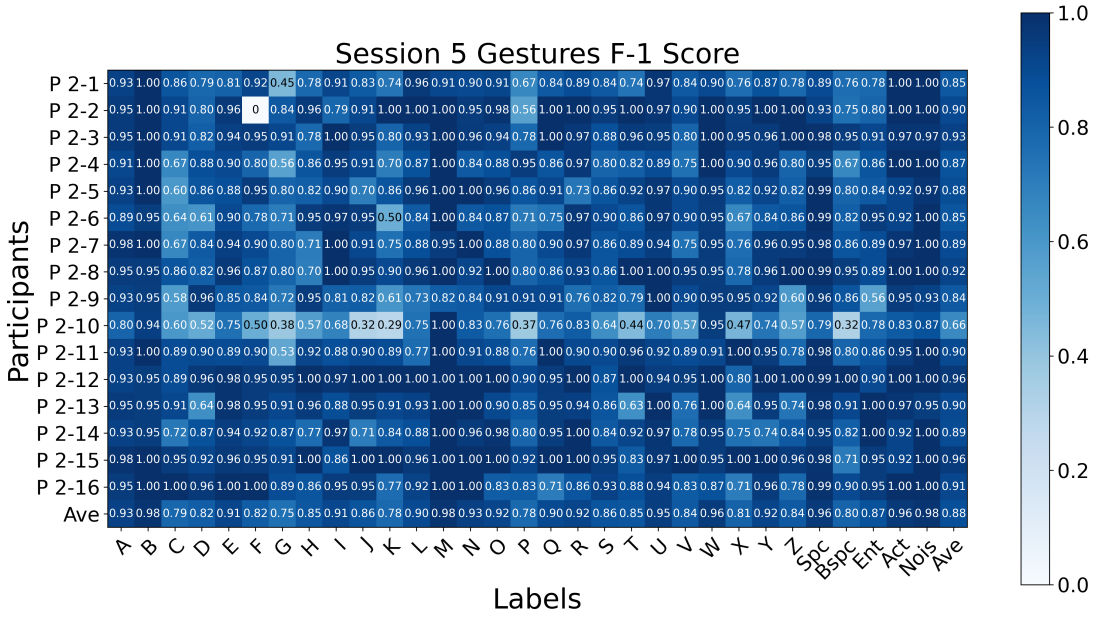

Fig. 15. The F-1 score in session 5 for 31 classes for 16 participants with transfer learning. *Ave* on the Labels and Participants are the average over all classes and participants. Other abbreviations are *Spc*-Space, *Bspc*-Backspace, *Ent*-Enter, *Act*-Activation, *Nois*-Noise. The number at the right bottom corner is the overall average F-1 score 0.883.

Figure 14 also shows that participant P2-10 under-performs compared to the average, having a z-score of -3.34 at the last session for the transfer model. Despite the fact they made an interesting observation, "*I think the easiest way to learn is just imagine my thumb is a piece of paper and I am writing on it*", which aligns with the design idea of *IndexPen*. In the post-session survey, this participant is the only user who noted that their hands were getting sore from performing the gesture for a long period of time. This shows that more consideration of comfort and ergonomics is essential. Fatigue could have a negative impact on the accuracy.

Additionally, we observed that *P2-1* achieved the highest accuracy in the third session; however, the performance dropped for the last two sessions. Similarly, we observed this dropping with other participants. (e.g., P2-1 (Session3-Session4), P2-9 (Session2-Session3)). Because we scheduled the experiment around participants' convenience, the gap between each session varied from one day to three weeks. We found that in most of the sessions that had a lower accuracy from the previous session, the participant was away for at least 1 week, and they usually required the experimenter to remind them of some gestures. This finding indicated that the gesture style could change over time within individuals, particularly with infrequent use [6].

According to the post session survey, there are two types of **challenging gestures** : complex/unnatural gestures and confusing gestures. For the first category, *Y*, *G*, and *O* were mentioned 14, 12, and six times respectively by different participants in different sessions. Those gestures either have complicated curves or unnatural movement.

P2-12 explained:

> *"It is like more movement and it is more trickier"*

However, the difficulty is not an essential factor that influences accuracy. *Y*, as the most frequently mentioned hard gesture, achieved 91.9% accuracy (9th highest accuracy over 30 gestures). In contrast, C was mentioned

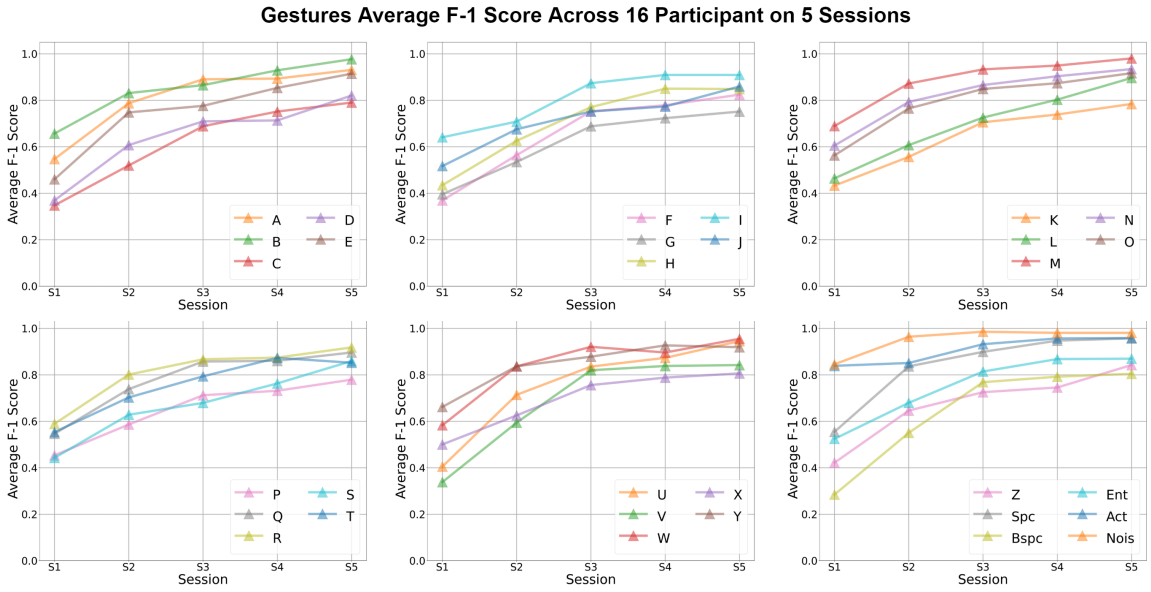

Fig. 16. Average F-1 score across all participants for each character. For clearer visualization, we plot 5 gestures on each sub-figure.

only once over the entire study, but it had the fourth lowest accuracy at the last session (79.0%). This is because complex gestures usually have more features that are more distinguishable comparing with simple gestures.

K, *Z*, and *F*, were mentioned 11, eight, and seven times. The most frequent reason was the direction is confusing. During the first and second sessions, participant usually did them in reverse (i.e., do the F as a T.). This problem has been referred to as as *Mirror-image confusion*, and *principal axis reflection* [17] and occurred more often than all other error types in participants. *P2-9* says:

> *"Some of them are harder because of their multiple strokes and their directions. I sometimes got confused about the direction of the gestures, either from inner index finger or from the outer index finger."*

This problem should be explored further to design the most understandable tutorial.

Fifteen out of sixteen users reported increasing **ease and comfort** using the *IndexPen* application as the model continuously adapted to their gesture style and as the user learned to optimize the hand posture for best detection result.

Fourteen participants reported having a hard time during the tutorial before the first session. The most common problem, as we mentioned in 6.5.5, was doing the gestures in reverse. However, all the participants felt that it became easy after the first session and the number of mistakes caused by writing error (not prediction) reduced from 1.58% to 0.49% on the second session. The following feedback on the second session reflects the learning curve:

> *P2-2:"Feels more comfortable and can follow those gesture very easily."*
> *P2-5:"Easier to write and follow. Now I have time to look at the input box at the top"*
> *P2-9:"I feel much more comfortable with writing the letters using my hands than the last time."*

Four participants suggested that the learning curve may be impacted by experience playing an instrument or similar activities requiring finger dexterity. In our experiment, eleven participants reported having engaged

in activities that require finger dexterity. However, we did not find a strong correlation between the learning curve and those activities. Excluding *P2-10*, the accuracy for the participants that did not report engaging in finger dexterity activities ($P2-4, 5, 6, 8$) had an accuracy of 51.4% in the first session and ended with 88.1% on the last session, which is in line with the average across all participants However, the correlation between finger dexterity activity and learning curve should be explored further on a larger population.

7 LIMITATIONS AND FUTURE WORK

The proposed *IndexPen* system achieved high accuracy across 30 gestures in addition to the *noise* class. However, the calibration procedure still requires more than 20 samples per character to obtain a reasonably high accuracy on a new user, and this is tedious for the end user. Further, additional people may introduce variance to the data and the classification model. On the other hand, training on a larger group of users and capturing more of the variability in the training set can lead to higher generalizability of the model. In addition, data augmentation [53] can have similar effects by artificially creating additional training data based on the existing data, leading to greater generalizability. Additionally, the gesture location could be an essential factor that affects accuracy. In future work, we would like to experiment with how the gesture accuracy varies at different angles and distances relative to the radar.

Given the successful classification of *IndexPen* text, we presume that similar gestures can be fitted to alphabets in addition to, or in place of English, and possibly extend to characters from ideographic languages such as Chinese and Japanese. We had participants who speak Spanish, French, Chinese, and Japanese. Participants thought extending *IndexPen* to Spanish and French could be easy since the characters are overlapping. For languages such as Chinese or Japanese, *IndexPen* could enable direct input of the characters, instead of requiring phonetic alphabets which are often used for input on devices with Qwerty keyboards. A related future direction is to include the numbers in the gesture space without compromising the discriminability of the existing gestures. Care would have to be given to the gesture design of characters with similar strokes (e.g. number "1" with the letter "I" and number "2" with the letter "z"). Applying a language model could help with differentiating similar gestures by giving greater weight to characters that would be more likely, given the context of the previous string of characters. This could also enable auto-correction and predictive writing, similar to what is found in the swipe keyboard on smartphones and smartwatches [52]. The additional layer of cross-letter and cross-word processing could improve the match ratio between what the user wants to write and what the system detects. It would be worth investigating to what size an *IndexPen* gesture set could go before the accuracy would decline.

IndexPen also has the potential to be used as a touchless interface in real-world applications such as ticketing machines, kiosk terminals, gas stations, elevator buttons, and medical settings for hygienic reasons. The gesture alphabet could be reduced for these specific applications (i.e., only open-door, close-door, and floor numbers in an elevator) and a higher precision would be expected from a smaller gesture set. *P2-5* suggested that " *this could be helpful for blind people to interact with other devices.*" *P2-15* suggested that *"This could be useful in video games to bring a more immersive experience. (Potentially VR)"*. These are a few areas where we could envision expanding the concepts explored here with *IndexPen*.

8 CONCLUSION

This is the first paper to demonstrate the feasibility of a high accuracy radar-based interaction technique for input of the entire English alphabet using 30 in-air, two-finger micro-gestures, designed to be easy to learn and remember because it is based on everyday hand-writing. In addition to demonstrating high accuracy recognition of more gestures than previous work (30 vs 11), our paradigm is based on fingertip-only gestures, using two fingers, and does not involve full hand swipes or larger gestures. It is also not constrained to personal used like the wearable devices; the system can potentially replace universal public input interfaces such as elevator buttons.

The proof-of-concept demonstration and feasibility studies provide insight for future HCI using mmWave radar, as it provides touch-free, private, familiar text input, that could extend to other languages.

ACKNOWLEDGMENTS

We would like to express our gratitude to the Computer Science department and Electrical and Computer Engineering department of Worcester Polytechnic Institute (WPI) for letting us run the user studies. Thanks to all user study participants from WPI. We are grateful to Manxueying Li, a Columbia University colleague who shared her expertise in constructing illustration figures. We thank four anonymous reviewers. We also thank Yining Hua for her contribution to the original software pipeline, and Zhenyuan Lei and An Yan for their contribution to the experiment. This work was supported in part by the Center for Wireless Information Network Studies (CWINS) Lab, Human-Computer Interaction (HCI) Lab at WPI and Laboratory for Intelligent Imaging and Neural Computing (LIINC) at Columbia University.

REFERENCES

[1] Heba Abdelnasser, Moustafa Youssef, and Khaled A Harras. 2015. Wigest: A ubiquitous wifi-based gesture recognition system. In *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 1472–1480.

[2] Ibrahim Alnujaim, Hashim Alali, Faisal Khan, and Youngwook Kim. 2018. Hand gesture recognition using input impedance variation of two antennas with transfer learning. *IEEE Sensors Journal* 18, 10 (2018), 4129–4135.

[3] Amin Arbabian, Steven Callender, Shinwon Kang, Mustafa Rangwala, and Ali M Niknejad. 2013. A 94 GHz mm-wave-to-baseband pulsed-radar transceiver with applications in imaging and gesture recognition. *IEEE Journal of Solid-State Circuits* 48, 4 (2013), 1055–1071.

[4] Maryam Asadi-Aghbolaghi, Albert Clapes, Marco Bellantonio, Hugo Jair Escalante, Víctor Ponce-López, Xavier Baró, Isabelle Guyon, Shohreh Kasaei, and Sergio Escalera. 2017. A survey on deep learning based approaches for action and gesture recognition in image sequences. In *2017 12th IEEE international conference on automatic face & gesture recognition (FG 2017)*. IEEE, 476–483.

[5] Sebastian Bock and Martin Weiß. 2019. A proof of local convergence for the Adam optimizer. In *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[6] Fazli Can and Jon M Patton. 2004. Change of writing style with time. *Computers and the Humanities* 38, 1 (2004), 61–82.

[7] Edwin Chan, Teddy Seyed, Wolfgang Stuerzlinger, Xing-Dong Yang, and Frank Maurer. 2016. User Elicitation on Single-hand Microgestures. 3403–3414. https://doi.org/10.1145/2858036.2858589

[8] Liwei Chan, Rong-Hao Liang, Ming-Chang Tsai, Kai-Yin Cheng, Chao-Huai Su, Mike Y Chen, Wen-Huang Cheng, and Bing-Yu Chen. 2013. FingerPad: private and subtle interaction using fingertips. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*. 255–260.

[9] Ke-Yu Chen, Shwetak N Patel, and Sean Keller. 2016. Finexus: Tracking precise motions of multiple fingertips using magnetic sensing. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 1504–1514.

[10] Dan C Cireşan, Ueli Meier, and Jürgen Schmidhuber. 2012. Transfer learning for Latin and Chinese characters with deep neural networks. In *The 2012 international joint conference on neural networks (IJCNN)*. IEEE, 1–6.

[11] David A Cook. 2014. The value of online learning and MRI: finding a niche for expensive technologies. *Medical teacher* 36, 11 (2014), 965–972.

[12] Ulysse Côté-Allard, Cheikh Latyr Fall, Alexandre Drouin, Alexandre Campeau-Lecours, Clément Gosselin, Kyrre Glette, François Laviolette, and Benoit Gosselin. 2019. Deep learning for electromyographic hand gesture signal classification using transfer learning. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 27, 4 (2019), 760–771.

[13] 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.

[14] Fatemeh Fahimi, Zhuo Zhang, Wooi Boon Goh, Tih-Shi Lee, Kai Keng Ang, and Cuntai Guan. 2019. Inter-subject transfer learning with an end-to-end deep convolutional neural network for EEG-based BCI. *Journal of neural engineering* 16, 2 (2019), 026007.

[15] Yun Fu and Thomas S Huang. 2007. hMouse: Head tracking driven virtual computer mouse. In *2007 IEEE Workshop on Applications of Computer Vision (WACV'07)*. IEEE, 30–30.

[16] Jun Gong, Yang Zhang, Xia Zhou, and Xing-Dong Yang. 2017. Pyro: Thumb-tip gesture recognition using pyroelectric infrared sensing. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. ACM, 553–563.

[17] Emma Gregory, Barbara Landau, and Michael McCloskey. 2011. Representation of object orientation in children: Evidence from mirror-image confusions. *Visual cognition* 19, 8 (2011), 1035–1062.

[18] Eiji Hayashi, Jaime Lien, Nicholas Gillian, Leonardo Giusti, Dave Weber, Jin Yamanaka, Lauren Bedal, and Ivan Poupyrev. 2021. RadarNet: Efficient Gesture Recognition Technique Utilizing a Miniature Radar Sensor. In *Proceedings of the 2021 CHI Conference on Human Factors*

in Computing Systems. 1–14.
[19] Wilbert Jan Heeringa. 2004. *Measuring dialect pronunciation differences using Levenshtein distance*. Ph. D. Dissertation. University Library Groningen][Host].
[20] Chen-Yu Hsu, Yuchen Liu, Zachary Kabelac, Rumen Hristov, Dina Katabi, and Christine Liu. 2017. Extracting gait velocity and stride length from surrounding radio signals. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2116–2126.
[21] J Stuart Hunter. 1986. The exponentially weighted moving average. *Journal of quality technology* 18, 4 (1986), 203–210.
[22] Cesar Iovescu and Sandeep Rao. 2017. The fundamentals of millimeter wave sensors. *Texas Instruments, SPYY005* (2017).
[23] Robert JK Jacob, Audrey Girouard, Leanne M Hirshfield, Michael S Horn, Orit Shaer, Erin Treacy Solovey, and Jamie Zigelbaum. 2008. Reality-based interaction: a framework for post-WIMP interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 201–210.
[24] Mohita Jaiswal, Vaidehi Sharmay, Abhishek Sharmaz, and Raghuvir Tomar. 2020. Transfer Learning with L2 Norm Regularization for classifying static Two Hand Hindi Sign Language Gestures. In *2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT)*. IEEE, 44–48.
[25] Vinay Jayaram, Morteza Alamgir, Yasemin Altun, Bernhard Scholkopf, and Moritz Grosse-Wentrup. 2016. Transfer learning in brain-computer interfaces. *IEEE Computational Intelligence Magazine* 11, 1 (2016), 20–31.
[26] Jing Jia, Geng Tu, Xin Deng, Chuchu Zhao, and Wenlong Yi. 2019. Real-time hand gestures system based on leap motion. *Concurrency and Computation: Practice and Experience* 31, 10 (2019), e4898.
[27] Faheem Khan, Seong Kyu Leem, and Sung Ho Cho. 2017. Hand-based gesture recognition for vehicular applications using IR-UWB radar. *Sensors* 17, 4 (2017), 833.
[28] Eva Kollorz, Jochen Penne, Joachim Hornegger, and Alexander Barke. 2008. Gesture recognition with a time-of-flight camera. *International Journal of Intelligent Systems Technologies and Applications* 5, 3 (2008), 334.
[29] Chan F Lam and David Kamins. 1989. Signature recognition through spectral analysis. *Pattern Recognition* 22, 1 (1989), 39–44.
[30] Ziheng Li, Zhenyuan Lei, An Yan, Erin Solovey, and Kaveh Pahlavan. 2020. ThuMouse: A micro-gesture cursor input through mmWave radar-based interaction. In *2020 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, 1–9.
[31] Jaime Lien, Nicholas Gillian, M Emre Karagozler, Patrick Amihood, Carsten Schwesig, Erik Olson, Hakim Raja, and Ivan Poupyrev. 2016. Soli: Ubiquitous gesture sensing with millimeter wave radar. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 142.
[32] Haipeng Liu, Yuheng Wang, Anfu Zhou, Hanyue He, Wei Wang, Kunpeng Wang, Peilin Pan, Yixuan Lu, Liang Liu, and Huadong Ma. 2020. Real-time arm gesture recognition in smart home scenarios via millimeter wave sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 4 (2020), 1–28.
[33] Haipeng Liu, Anfu Zhou, Zihe Dong, Yuyang Sun, Jiahe Zhang, Liang Liu, Huadong Ma, Jianhua Liu, and Ning Yang. 2021. M-Gesture: Person-Independent Real-Time In-Air Gesture Recognition Using Commodity Millimeter Wave Radar. *IEEE Internet of Things Journal* (2021), 1–1. https://doi.org/10.1109/JIOT.2021.3098338
[34] Fabien Lotte and Cuntai Guan. 2010. Regularizing common spatial patterns to improve BCI designs: unified theory and new algorithms. *IEEE Transactions on biomedical Engineering* 58, 2 (2010), 355–362.
[35] ZN Low, JH Cheong, and CL Law. 2005. Low-cost PCB antenna for UWB applications. *IEEE antennas and wireless propagation letters* 4 (2005), 237–239.
[36] Chris Xiaoxuan Lu, Muhamad Risqi U Saputra, Peijun Zhao, Yasin Almalioglu, Pedro PB de Gusmao, Changhao Chen, Ke Sun, Niki Trigoni, and Andrew Markham. 2020. milliEgo: mmWave Aided Egomotion Estimation with Deep Sensor Fusion. *arXiv preprint arXiv:2006.02266* (2020).
[37] Robin Lu and Yan Zhang. 2013. Balanced debounce circuit with noise filter for digital system. US Patent 8,502,593.
[38] Yongsen Ma, Gang Zhou, Shuangquan Wang, Hongyang Zhao, and Woosub Jung. 2018. Signfi: Sign language recognition using wifi. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 1 (2018), 1–21.
[39] Giulio Marin, Fabio Dominio, and Pietro Zanuttigh. 2014. Hand gesture recognition with leap motion and kinect devices. In *2014 IEEE International conference on image processing (ICIP)*. IEEE, 1565–1569.
[40] Pavlo Molchanov, Shalini Gupta, Kihwan Kim, and Jan Kautz. 2015. Hand gesture recognition with 3D convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 1–7.
[41] Oyebade K Oyedotun and Adnan Khashman. 2017. Deep learning in vision-based static hand gesture recognition. *Neural Computing and Applications* 28, 12 (2017), 3941–3951.
[42] Kaveh Pahlavan, Julang Ying, Ziheng Li, Erin Solovey, John Loftus, and Zehua Dong. 2020. RF Cloud for Cyberspace Intelligence. *IEEE Access* (2020).
[43] Sameera Palipana, Dariush Salami, Luis A Leiva, and Stephan Sigg. 2021. Pantomime: Mid-air gesture recognition with sparse millimeter-wave radar point clouds. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 1 (2021), 1–27.
[44] Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2009), 1345–1359.

[45] Qifan Pu, Sidhant Gupta, Shyamnath Gollakota, and Shwetak Patel. 2013. Whole-home gesture recognition using wireless signals. In *Proceedings of the 19th annual international conference on Mobile computing & networking*. 27–38.

[46] Qifan Pu, Sidhant Gupta, Shyamnath Gollakota, and Shwetak Patel. 2015. Gesture recognition using wireless signals. *GetMobile: Mobile Computing and Communications* 18, 4 (2015), 15–18.

[47] Edmar Rezende, Guilherme Ruppert, Tiago Carvalho, Fabio Ramos, and Paulo de Geus. 2017. Malicious Software Classification Using Transfer Learning of ResNet-50 Deep Neural Network. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 1011–1014. https://doi.org/10.1109/ICMLA.2017.00-19

[48] Herbert Richter. 2000. Palm pilot holder. US Patent App. 29/114,214.

[49] Dariush Salami, Ramin Hasibi, Sameera Palipana, Petar Popovski, Tom Michoel, and Stephan Sigg. 2021. Tesla-Rapture: A Lightweight Gesture Recognition System from mmWave Radar Point Clouds. *arXiv preprint arXiv:2109.06448* (2021).

[50] T Scott Saponas, Desney S Tan, Dan Morris, and Ravin Balakrishnan. 2008. Demonstrating the feasibility of using forearm electromyography for muscle-computer interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 515–524.

[51] T Scott Saponas, Desney S Tan, Dan Morris, Ravin Balakrishnan, Jim Turner, and James A Landay. 2009. Enabling always-available input with muscle-computer interfaces. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*. 167–176.

[52] Yuan-Fu Shao, Masatoshi Chang-Ogimoto, Reinhard Pointner, Yu-Chih Lin, Chen-Ting Wu, and Mike Chen. 2016. SwipeKey: a swipe-based keyboard design for smartwatches. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*. 60–71.

[53] Connor Shorten and Taghi M Khoshgoftaar. 2019. A survey on image data augmentation for deep learning. *Journal of Big Data* 6, 1 (2019), 1–48.

[54] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[55] Karly A Smith, Clément Csech, David Murdoch, and George Shaker. 2018. Gesture recognition using mm-wave sensor for human-car interface. *IEEE sensors letters* 2, 2 (2018), 1–4.

[56] Samuel L Smith, Pieter-Jan Kindermans, Chris Ying, and Quoc V Le. 2017. Don't decay the learning rate, increase the batch size. *arXiv preprint arXiv:1711.00489* (2017).

[57] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[58] Srikanth Tammina. 2019. Transfer learning using vgg-16 with deep convolutional neural network for classifying images. *International Journal of Scientific and Research Publications (IJSRP)* 9, 10 (2019), 143–150.

[59] Sheng Tan, Linghan Zhang, Zi Wang, and Jie Yang. 2019. MultiTrack: Multi-user tracking and activity recognition using commodity WiFi. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–12.

[60] Jilin Tu, Thomas Huang, and Hai Tao. 2005. Face as mouse through visual face tracking. In *The 2nd Canadian Conference on Computer and Robot Vision (CRV'05)*. IEEE, 339–346.

[61] Saiwen Wang, Jie Song, Jaime Lien, Ivan Poupyrev, and Otmar Hilliges. 2016. Interacting with soli: Exploring fine-grained dynamic gesture recognition in the radio-frequency spectrum. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, 851–860.

[62] Yao Wang and Qin-Fan Zhu. 1998. Error control and concealment for video communication: A review. *Proc. IEEE* 86, 5 (1998), 974–997.

[63] Frank Weichert, Daniel Bachmann, Bartholomäus Rudak, and Denis Fisseler. 2013. Analysis of the accuracy and robustness of the leap motion controller. *Sensors* 13, 5 (2013), 6380–6393.

[64] Xiaoling Xia, Cui Xu, and Bing Nan. 2017. Inception-v3 for flower classification. In *2017 2nd International Conference on Image, Vision and Computing (ICIVC)*. IEEE, 783–787.

[65] Xiao-Ling Xia, Cui Xu, and Bing Nan. 2017. Facial expression recognition based on tensorflow platform. In *ITM Web of Conferences*, Vol. 12. EDP Sciences, 01005.

[66] 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*. 883–899.

[67] Hui-Shyong Yeo, Gergely Flamich, Patrick Schrempf, David Harris-Birtill, and Aaron Quigley. 2016. Radarcat: Radar categorization for input & interaction. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. 833–841.

[68] Jih-Tsun Yu, Li Yen, and Po-Hsuan Tseng. 2020. mmWave radar-based hand gesture recognition using range-angle image. In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*. IEEE, 1–5.

[69] Yu Zhang, Guoxu Zhou, Jing Jin, Minjue Wang, Xingyu Wang, and Andrzej Cichocki. 2013. L1-regularized multiway canonical correlation analysis for SSVEP-based BCI. *IEEE transactions on neural systems and rehabilitation engineering* 21, 6 (2013), 887–896.

[70] Yang Zhang, Junhan Zhou, Gierad Laput, and Chris Harrison. 2016. Skintrack: Using the body as an electrical waveguide for continuous finger tracking on the skin. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 1491–1503.

[71] Chen Zhao, Ke-Yu Chen, Md Tanvir Islam Aumi, Shwetak Patel, and Matthew S Reynolds. 2014. SideSwipe: detecting in-air gestures around mobile devices using actual GSM signal. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. 527–534.

[72] Yongpan Zou, Jiang Xiao, Jinsong Han, Kaishun Wu, Yun Li, and Lionel M Ni. 2016. Grfid: A device-free rfid-based gesture recognition system. *IEEE Transactions on Mobile Computing* 16, 2 (2016), 381–393.

A APPENDIX

A.1 Confusion Matrix

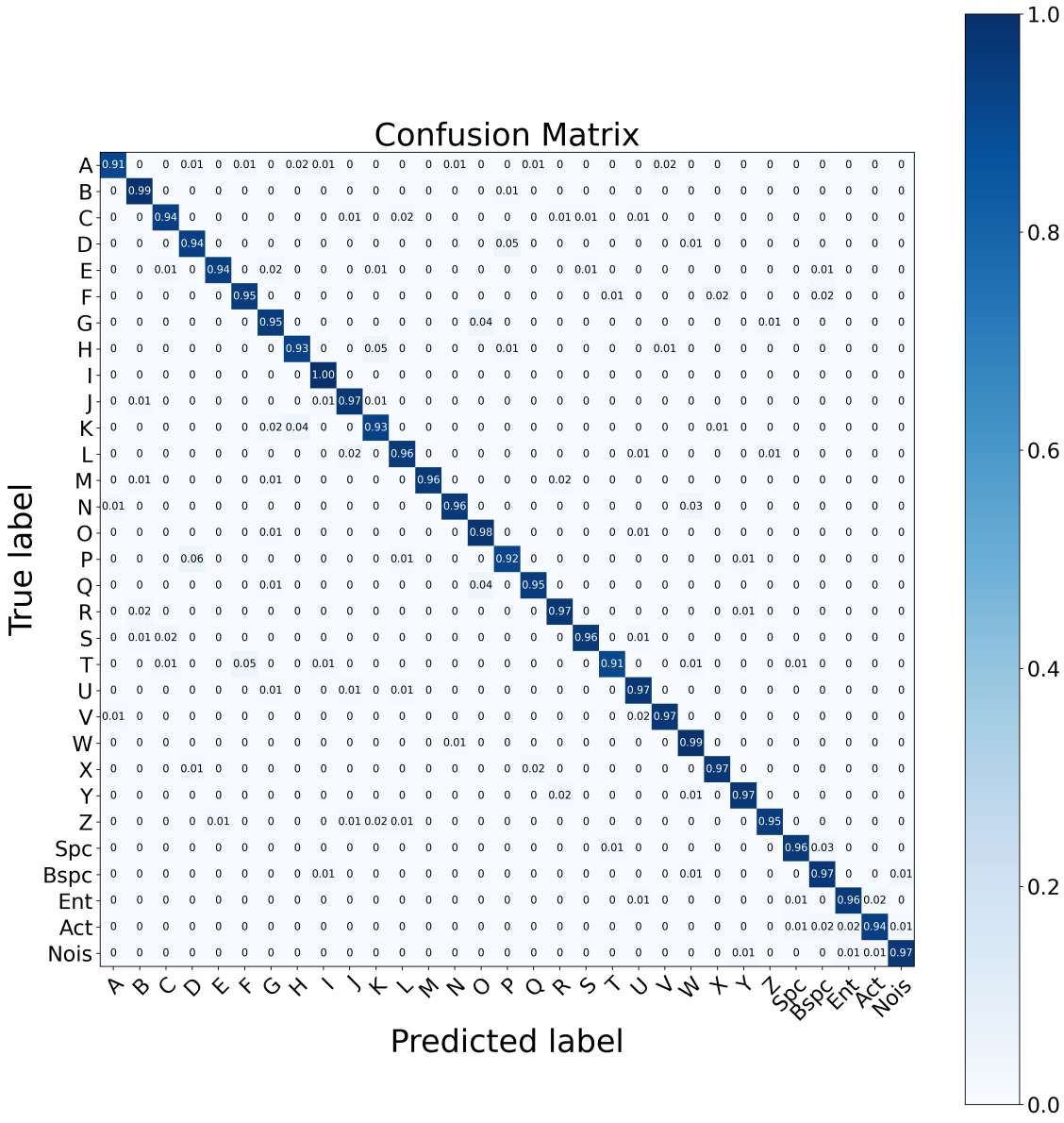

Fig. 17. The validation confusion matrix from user study 1 (section 5). The gestures with similar signature have more likely to be confused (i.e., between D & P, T & F).

A.2 Ablation Study

Figure 18 shows the result from five ablation studies. Figure 18(a) represents the loss and accuracy without the dropout layer. The final test accuracy for without dropout layer is 95.5% which is similar to 95.6% for adding dropout layers.

The final test accuracy for using only RD profile (without angular information from RAE) is 92.8%, compared to 95.6% of using both. We also see a slightly larger generalization gap when using only RD, where the test accuracy tilts upward as the epoch grows. Similarly, using only RAE (ignoring the information on the speed of movement), training with early stopping gives an accuracy of 63.4%. The relatively poor performance indicates that RD contains important information for decoding the gesture. At the same time, RAE can supplement the classification but is not sufficient as the sole feature for the model.

In the fourth and fifth ablation study, we justify our approach in creating separate convolutional layers for RD and RAE by showing the contrary: we assume that RD and RAE have the same probability prior, to be learned by the same convolution layers. With it, we combine RD and RAE along the horizontal axis (horizontal axis is the velocity in RD profile and RAE, it is azimuth and elevation) with input size to the network equal to $(8, 16 + 64, 1)$. Similarly, we combine RD and RAE along the vertical axis (vertical axis is the range for both RD and RAE, see figure 4 and 5). We resize the RD from $(8, 16, 1)$ to $(8, 64, 1)$ and the input dimension is $(8, 64, 2)$. In both cases, the test losses and accuracies are not stable, showing that RAE and RD have different spatial distributions that need to be optimized independently during training.

A.3 Experiment Software

We created three applications (Figures 20, 21, and 19) to enable the user study. Figure 20 is the data recording interface that captures video stream, radar image stream, and event marker from stimulus presentation software 19 with synchronized timestamps.

We present the real-time text input box and instruction images to the user during the study as shown in Figure 19. The displayed image includes the gesture that the participant needs to write as well as the instruction shown in blue lines. The green dot is the starting position of the index finger on the thumb and the path is labeled with indexes. The progress bar shows the remaining time for that gesture, and participants were instructed to finish writing that gesture during this fixed interval (four seconds in our experiment).

The real-time inference application in Figure 21 receives the radar stream from the recording application and returns the predictions. The raw radar profiles and the profiles after clutter removal are shown on the top. The model takes clutter-free profiles for prediction and returns the temporal probability evolution as well as the instantaneous probability from predictions on the visualization panel on the left. We implemented the debouncing algorithm from Section 4.4 based on the prediction result to invoke keystrokes. Participants can adjust their writing style according to the temporal probability evolution graph between sentences for higher accuracy.

A.4 F-1 Score for Each Session

The gesture accuracy for each session is shown in Figure 22, 23,24, 25, and 15.

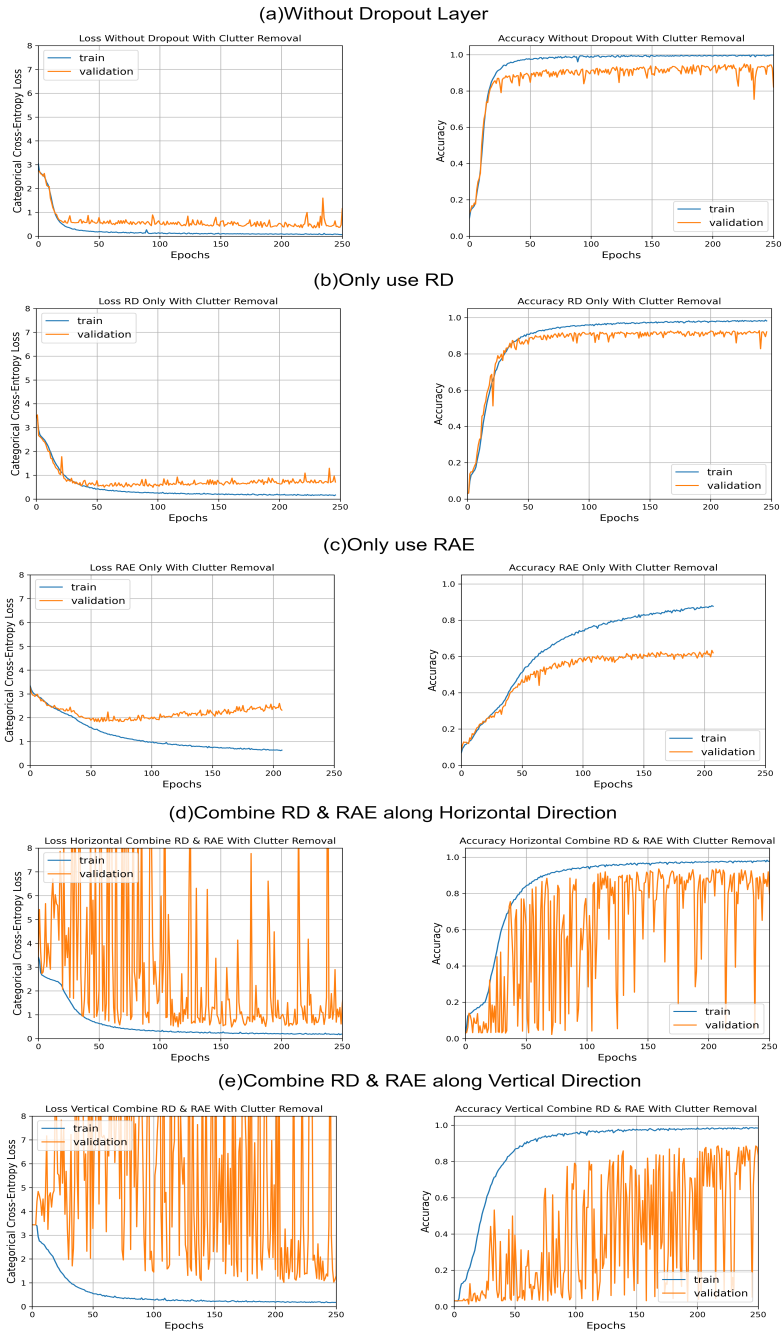

Fig. 18. The ablation study results. (a): Without Dropout Layer (b) Use RD as the only input to the neural network with clutter removal. (c) Use RAE as the only input to the neural network as the only input. (d): Stack RA and RAE along width axis and use the combined image as a single input to the network. (e) Up-sample RD from (8,16) to (8,64) and stack RD and RAE along depth axis.

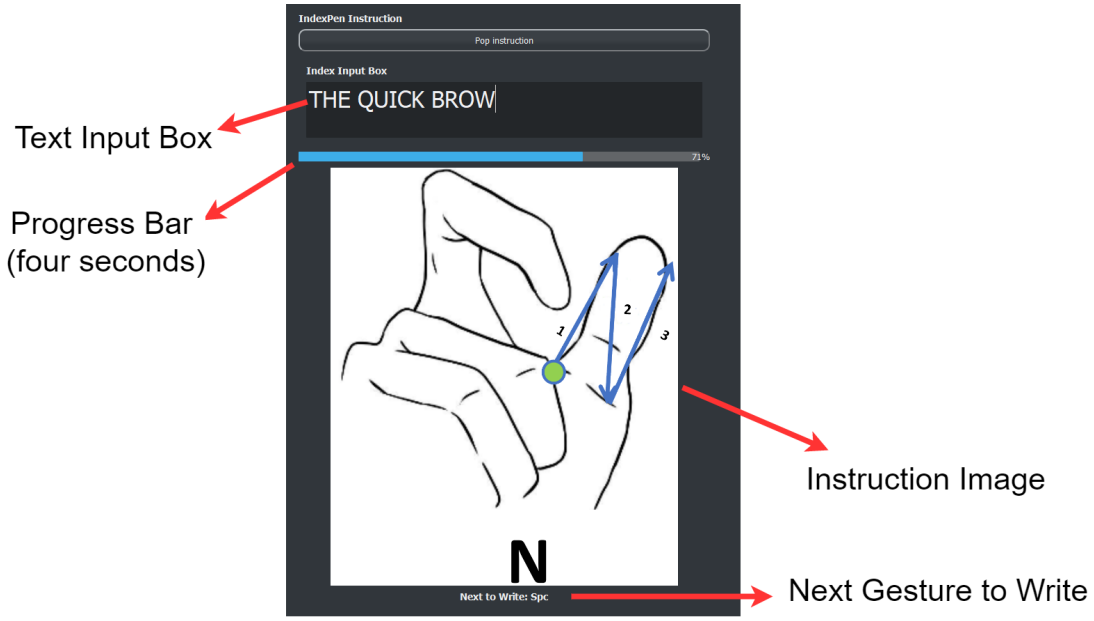

Fig. 19. A example participant was writing "THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG". Participants were instructed to perform the given gesture before the progress bar ends (four seconds). The text input shows the real-time inference result. The green dot on the gesture image indicates the starting position of index finger on thumb and the paths are labeled in numbers and arrows.

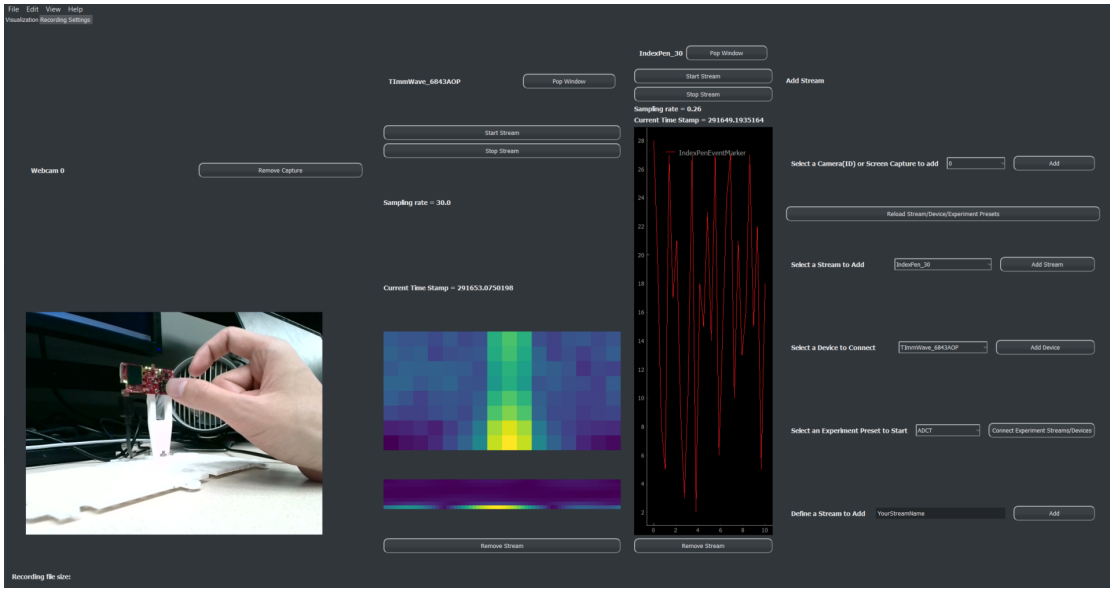

Fig. 20. The recording application that synchronize the video recording (on the left), radar profiles (in the middle), and event marker (on the right) that encoded into 31 integers between 1-31 from stimulation presentation software in Figure 19.

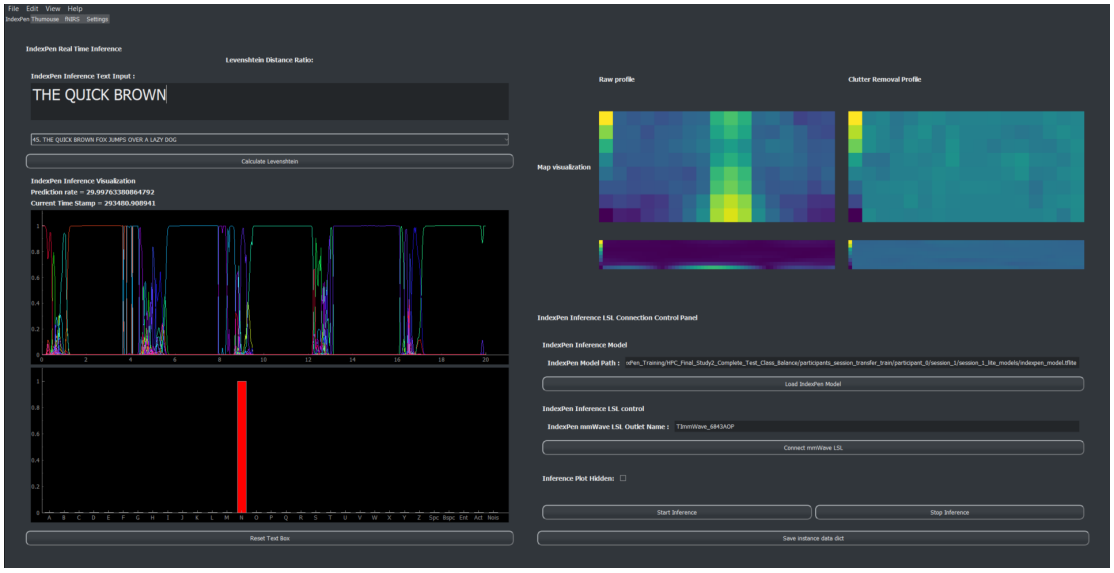

Fig. 21. Real-time inference application receives the radar profile streams from data recording application from Figure 20 and is processed with clutter removal algorithm. The visualization shows the temporal probability evolution and profile maps before and after clutter removal. The bar chart shows the instantaneous probability at that moment.

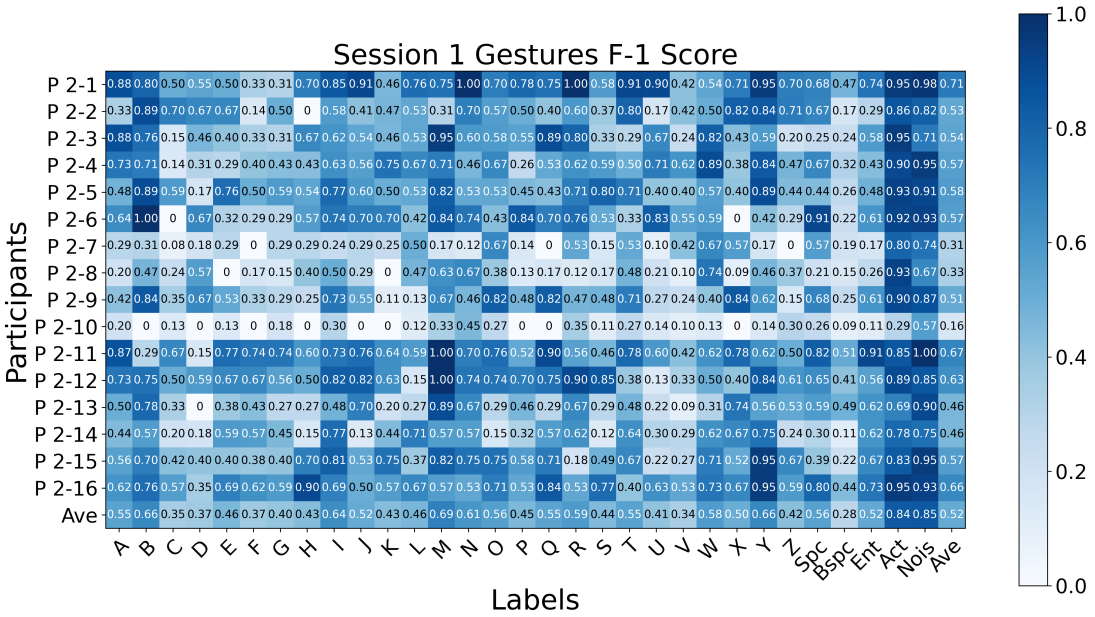

Fig. 22. User study first session gestures F1 score for each gesture for each participant.

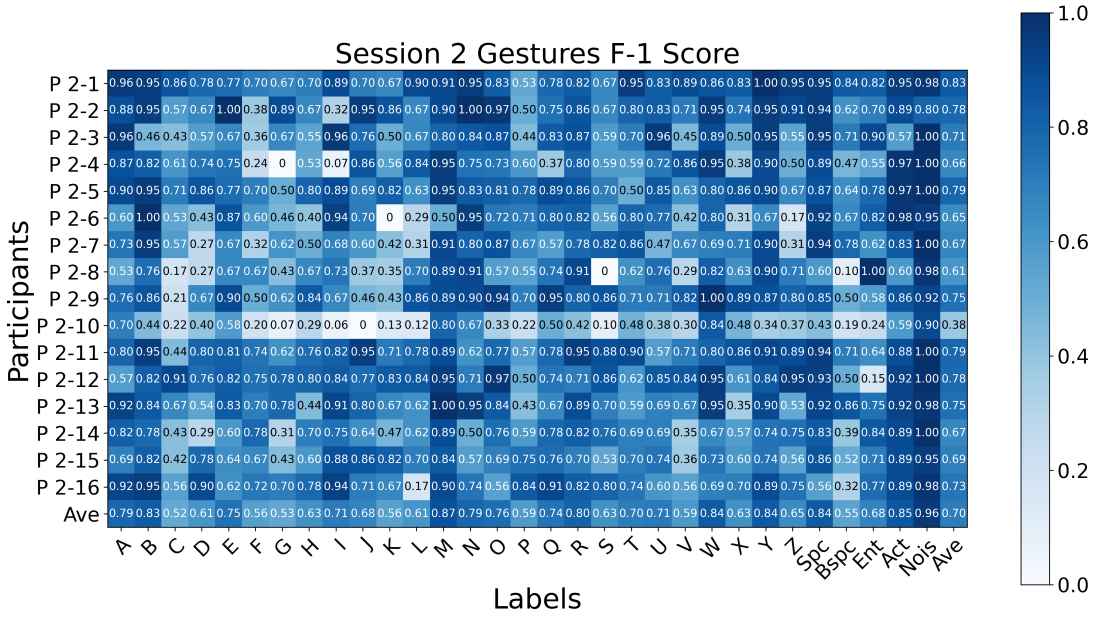

Fig. 23. User study second session gestures F1 score for each gesture for each participant.

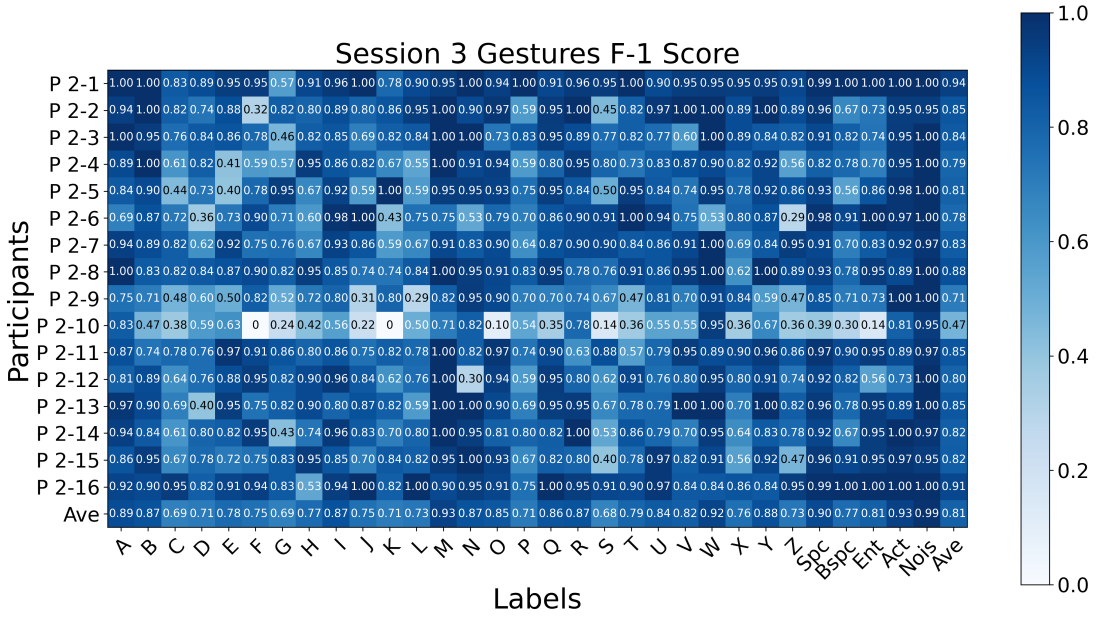

Fig. 24. User study third session gestures F1 score for each gesture for each participant.

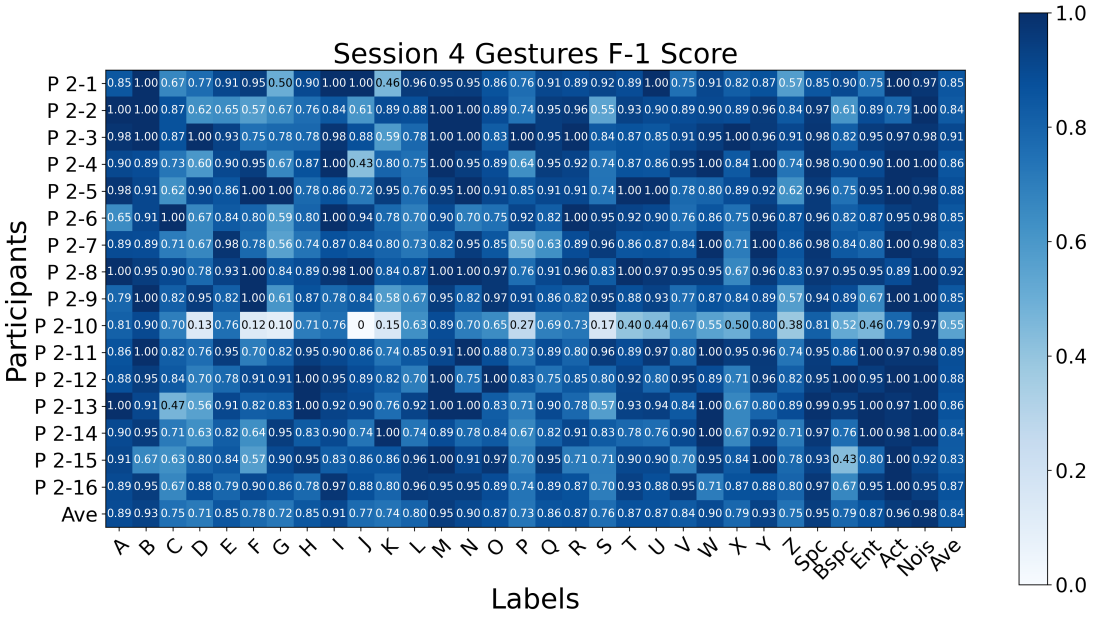

Fig. 25. User study fourth session gestures F1 score for each gesture for each participant.