Write, Attend and Spell: End-to-end Free-style Handwriting Recognition Using Smartwatches

***Abstract*—Text entry on a smartwatch is challenging due to its small form factor. Handwriting recognition using the built-in sensors of the watch (motion sensors, microphones, etc.) provides an efficient and natural solution to deal with this issue. However, prior works mainly focus on individual letter recognition rather than word recognition. Therefore, they need users to pause between adjacent letters for segmentation, which is counter-intuitive and significantly decreases the input speed. In this paper, we present 'Write, Attend and Spell' (WriteAS), a word-level text-entry system which enables end-to-end free-style handwriting recognition using the motion signals of the smartwatch. First, we combine a multimodal convolutional neural network (CNN) and a Gate Recurrent Unit (GRU) to abstract informative features across modalities and time. Then, we exploit a multi-task encoder-decoder network to get around letter segmentation and output words in an end-to-end way. We construct the first sequence-to-sequence handwriting dataset using smartwatch. WriteAS can yield 8.1% character error rate (CER) on 240 words for new users and 9.7% CER for words unseen in the training set. Given its promising performance, we envision that WriteAS can be a fast and accurate input tool for smartwatch.**

***Index Terms*—smartwatch, text entry, handwriting, end-to-end**

I. Introduction

Smartwatch is emerging as the most popular wearable devices in recent years. It offers great convenience to users through a wide range of on-the-go applications such as checking emails, writing notes, messaging, and social networking. Text entry is indispensable to these applications. Unfortunately, efficient text entry on smartwatch is still challenging. Touch input, as the primary input form, is limited by the small screen size which makes it much slower than on smartphones [1]. Systems aiming to facilitate touch input, such as multi-tap [2] and finger identification [3], impose burdens on users to adapt to new touch methods. Although voice input can be a promising alternative, it is not suitable in some public settings due to the social acceptability (e.g., at meetings) and noise interference [4].

In this paper, we turn our attention to another traditional input method, handwriting. The powerful computational capabilities and versatile sensors of smartwatch make it possible to recognize handwriting in air or on a surface. Compared with other text entry methods, handwriting input is more natural and expressive. There has been a rich body of research about handwriting recognition on mobile devices. These methodologies mainly fall into two categories in terms of their sensing modality, i.e., acoustic-based and sensor-based. Acoustic-based methods enable input by capturing the subtle sound generated by the pen/finger writing on the table [5] [6] [7] [8]. These methods, however, can only work in quiet indoor environments with minimum ambient noise. Prior studies have shown that people feel no compulsion to wear the smartwatch on the dominate hand if they can get better services (e.g., exercise feedback) [9]. Therefore, sensor-based methods mainly leverage the inertial sensors in the watch to monitor hand or arm movements and thereby enable writing recognition. For example, AirContour [10] converts arm movements to contours in 3D space to recognize characters. SHOW [9] uses the elbow as the support point to deduce users' handwriting on surfaces. It has a much lower error rate than the touch-input method and achieves comparable input efficiency. Some other researches use the sensors embedded in a pen to enable interaction with mobile devices, such as MagHacker [11] based on the magnet and Pentelligence [12] combing inertial sensors and microphones together into a pen.

Nevertheless, state-of-the-art methods only focus on individual letter recognition rather than word recognition. They need users to pause between two adjacent letters for temporal segmentation. This requirement significantly degrades user experience and the input speed since people tend to write in a cursive and continuous way, where a natural pause is usually at the word level. In addition, the letter segmentation itself is non-trivial and inevitably propagates errors into subsequent steps [8]. One possible solution is to regard word as the basic classification unit and directly output word labels. While theoretically plausible, it is impractical to collect the data of such a huge vocabulary (English words in our paper), let alone many undefined words (e.g., name). Another shortcoming of existing works is that they are vulnerable to complex writing behaviors. For example, most methods build the user-dependent model which means their performance will degrade severely when a new user comes. Moreover, some works [5] [6] can only deal with the print-style capital letters rather than the free-style lowercase input while the latter is much more frequently used in our daily life.

To deal with the above limitations, we propose 'Write, Attend and Spell' (WriteAS), a word-level text-entry system for smartwatch that enables free-style handwriting recognition. Specifically, WriteAS leverages multiple inertial sensors in the smartwatch, i.e., accelerometer, gyroscope and gravity sensor, to capture the hand movements of users who just need to write on a surface with their index fingers in their habitual way (as depicted in Fig. 1). Different from previous methods, WriteAS enables word-level recognition in an end-to-end way which means that it does not need letter segmentation but can directly map the motion signals into words while the basic modeling

Fig. 1. WriteAS: the user handwrites on a surface with the index finger and the smartwatch will display the recognized results on the screen.

units are still the 26 English letters. In addition, WriteAS can generalize very well to various users and writing behaviors.

We empower WriteAS through three major components. First, considering that different sensors have different granularity, representation capabilities and value distributions, we elaborate a multimodal Convolutional Neural Networks (CNN) to learn local features of each modality and global features across modalities. Second, we treat handwriting recognition as a sequence modeling problem [13] and adopt a multi-task encoder-decoder network to directly output words in an end-to-end way without letter segmentation. Specifically, in our multi-task framework, we leverage Connectionist Temporal Classification (CTC) [14] to guide the training of the attention-based encoder-decoder network [15]. This design clearly boosts the alignment performance between the motion signals and the output words while automatically learning the linguistic relations in words. Third, as writing speed, font size and watch positions may differ from person to person, we design a series of data augmentation mechanisms by correlating the motion signals with the writing variations, which can effectively avoid overfitting and cover different scenarios.

To evaluate the performance of WriteAS, we construct the first sequence-to-sequence handwriting dataset using smartwatch, which consists of 240 most commonly used English words and $>$ 20000 samples. These data is collected from 12 volunteers at different time and in different environments. We fine-tune each step in WriteAS to provide a strong baseline for the future study in this field. WriteAS can achieve 0.6% character error rate (CER) under user-dependent test and 8.1% CER under user-independent test. For words not in the training set, the average CER is 9.7%. Note that the capability of recognizing unseen words eliminates the burden of collecting all words. We also implement an real-time app on the commercial smartwatch and compare it with the touch-input method through in-the-field evaluations. Results demonstrate that WriteAS outperforms touch input by a large margin in both accuracy and efficiency. We will release the dataset and code publicly for the research community.

The contributions of this paper are summarized as follows:

- We propose WriteAS, a first-of-its-kind end-to-end handwriting recognition system for smartwatch which is reliable and robust under different writing behaviors.
- We employ the multimodal-CNN and GRU to abstract representations containing intra-modality, cross-modality and temporal relations. Then, we adopt a multi-task encoder-decoder network to directly map motion signals into words, which gets around letter segmentation and allows users to write in a cursive and continuous way.
- We construct a high-quality sequence-to-sequence handwriting dataset including extensive word-level samples. Evaluation results show that WriteAS yields high recognition performance for both unseen users and words.

The rest of the paper is organized as follows: In Section II, we introduce the preliminaries of handwriting on smartwatch. In Section III, we present the details of WriteAS, including the signal preprocessing and the deep neutral networks. In Section IV, we provide the evaluations of WriteAS. In Section V, we review the related works. In Section VI, we make the conclusion of the paper.

II. Preliminary

A. Handwriting with a Smartwatch

When users write with smartwatch wearing on their dominate hands, they can use different support points, i.e., shoulder, elbow and wrist. The corresponding amplitudes of smartwatch movements decrease in turn. The shoulder is usually used for in-air writing and needs the movement of the arm [10]. Thus it is not suitable for handwriting input on smartwatch since whole-arm motions tend to make users exhausted. SHOW [9] leverages the elbow as the support point to balance comfort and accuracy. In WriteAS, we choose the wrist as the support point as it is consistent with people's writing behaviors in daily life. Moreover, our elaborate deep learning network can still extract stable and informative features in this scenario although the corresponding movements are subtle.

There are several available inertial sensors on a smartwatch that can capture human motions, including but not limited to accelerometers and gyroscopes. A gravity sensor is a low-power software sensor fusing both accelerometer and gyroscope [16] which can reflect subtle changes of wrist poses. We choose the data from these three sensors in our system. Fig. 2 is the accelerometer, gyroscope and gravity signals (DC components have been removed) when a volunteer writes letter 'a' with the index finger, from which we can observe a clear fluctuation pattern.

B. Word-level Handwriting Recognition

When it comes to word-level handwriting, prior research, e.g. WritingHacker [5], WordRecorder [6], WritingRecorder [8], and SHOW [9] require users to pause between adjacent letters and adopt a three-step pipeline to recognize words. First step is to perform letter segmentation so that each segment contains only one letter. Then, in the second step, they use classification methods to get the probability distribution of labels $P(x_i)$ in each segment. Finally, a language model

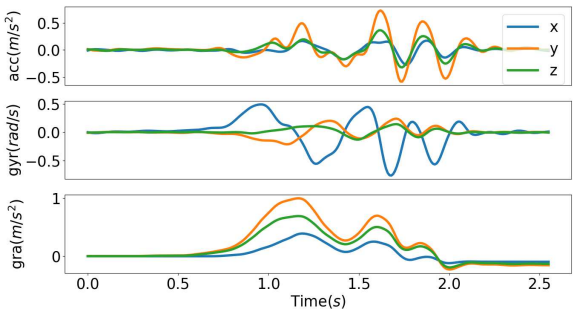

Fig. 2. Sensory data when handwriting the letter 'a'.

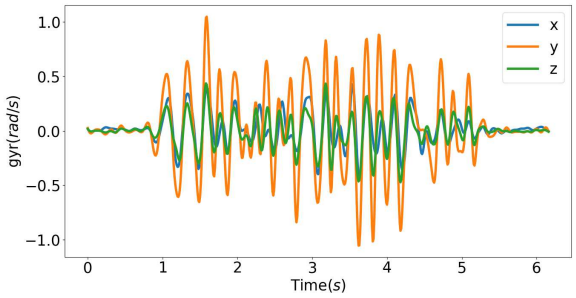

Fig. 3. Gyroscope signals when handwriting the word 'machine'.

will be introduced to infer the most likely word label, which satisfies:

$$argmaxL(x_1, x_2, \cdots, x_i, \cdots) * \prod_i P(x_i) \tag{1}$$

L is the probability of this word (letter sequence) under the context-aware model. The performance of these methods is highly dependent on the accuracy of the first step, and inaccurate segmentation will inevitably propagate errors into subsequent steps. Unfortunately, temporal letter segmentation is non-trivial because a shorter pause between letters is not enough for segmentation and a longer pause will notably slow down the input speed. Worse still, users usually pause between words rather than letters as they tend to write in a cursive way, which makes letter segmentation very difficult, if not impossible. Fig. 3 is the gyroscope data when a volunteer writes the word 'machine' in the free style. The word consists of 7 letters. But it is difficult to find the number of letters or temporal location of each letter in the figure. In WriteAS, we employ the encoder-decoder mechanism to enable word-level handwriting recognition in an end-to-end way, which removes the requirement of temporal segmentation.

III. System Design

In this section, we detail our system design. Fig. 4 illustrates the overview of WriteAS. First, the continuous signal stream is divided into word segments. In the training phase, the data augmentation mechanism is used to enlarge the data. Afterwards, the system employs a multimodal CNN to handle

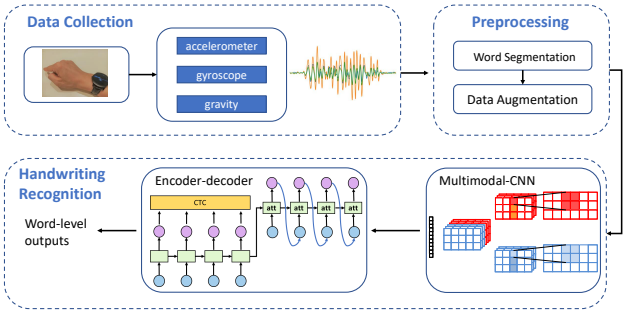

Fig. 4. Overview of WriteAS.

sensor heterogeneity and abstract informative features. Such features, then, are sent to a encoder-decoder network to output words.

A. Preprocessing

Word Segmentation. As mentioned in [6] [8] [9], there is usually a clear pause between adjacent words when people handwrite, and thus the word segmentation is much easier than letter segmentation. In WriteAS, we adopt a light-weight method, i.e., computing the energy level of motion signals, to detect the writing behavior. The sampling rate in our system is 10Hz, and we use a sliding window of 10 readings (50% overlap) to accumulate the energy:

$$A(t) = \sum_{k=t}^{t+10} s^2(k) \tag{2}$$

$$s(k) = \sqrt{x^2(k) + y^2(k) + z^2(k)} \tag{3}$$

where $x(k)$, $y(k)$ and $z(k)$ is the value of the gyroscope signal along the three axes, indexed by k. We choose the gyroscope data as it exhibits the most pronounced fluctuations when users handwrite. We compare $A(t)$ with an empirical threshold to identify the word-level boundaries of the motion signals. It is noteworthy that our system can still give an accurate output when the segmentation is wrong since our deep learning network is sequence-to-sequence and the modeling unit is letter rather than word.

Data Augmentation. The deep learning network usually has a huge demand for the training data. However, it is hard for us to collect a sufficient amount of handwriting quality data in practice. Data augmentation is a common strategy to deal with this problem. In this paper, we design our data augmentation mechanism taking into account the fact that the writing speed, font size and watch orientation differ from person to person.

Fig. 5 is the distributions of writing durations in our dataset (12 volunteers and $>$ 20000 samples) for the same word 'machine'. It can be observed that the durations are very disperse, ranging from 3.5s to 7.5s. To deal with the speed diversity, we exploit the idea of the time warping technique used in speech recognition [17] to enrich the training data. Specifically, given the motion signal $x(t)$, we expand or

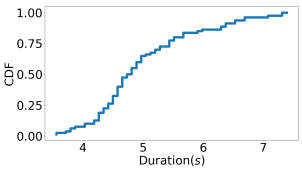

Fig. 5. The duration of writing the word 'machine'.

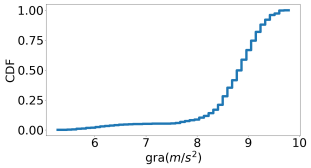

Fig. 6. The gravity data along the z axis when not writing.

contract it in the time axis by a factor α, thereby generating a new profile $x(\alpha t)$. Our another warping mechanism is to randomly select n ($n = 2$) segments of length W ($W = 0.5s$) from the data and warp it along the time axis. We also use the same method to generate the data when writing different letter size, in which we wrap the data along the amplitude axis.

Each time users wear the watch, the watch position, especially its facing direction, is not exactly the same. To illustrate this, we extract the gravity data when users pause between two adjacent words from our dataset. We calculate the mean value for each experiment and depict the distributions in Fig. 6. Although we ask users to try their best to put the watch in the same direction, it can be seen from the figure that the direction is still different each time. In WriteAS, we leverage the rotation matrix to generate new profiles in different facing directions:

$$x'(t) = R(\theta)x(t) \tag{4}$$

where $R(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta) & -sin(\theta) \\ 0 & sin(\theta) & cos(\theta) \end{bmatrix}$ is the rotation matrix of θ over the arm axis (x axis). Due to the wrist anatomy of human, the rotation angle is usually in $[-30°, 30°]$ [9]. Thus we generate 12 new profiles by changing θ from $-30°$ to $30°$ with a step of $5°$.

B. Multimodal CNN

As convolutional neural networks exhibit superior representation capability in image and text classification, we employ CNN to design the front-end. We take into account the property of the sensory data in designing the CNN. First, the signals of each inertial sensor are temporal signals 3 axes. So the CNN should aggregate information across time and axes. Second, the CNN model needs to fuse the information from the heterogeneous sensors, i.e., accelerometer, gyroscope and gravity. An intuitive idea is to directly concatenate the signals to form 9-channel inputs. However, this method regards both modalities share the same signal-noise patterns and representation capability, and thus incline to miss the important local interactions within each modality. Previous works [18] [19] have shown that intra-modality interactions are stronger than cross-modality ones. To deal with these issues, we employ a multimodal CNN to learn intra-modality and cross-modality representations hierarchically, as depicted in Fig. 7.

The extracted word-level signal stream is first segmented into a sequence of overlapping clips with length T. We adopt three individual branches to extract intra-modality features

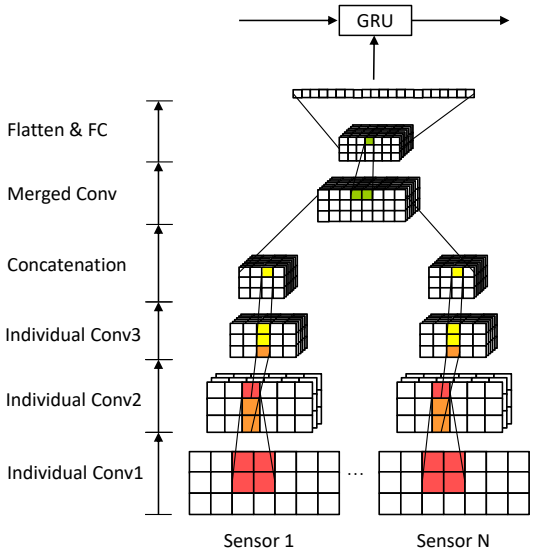

Fig. 7. The architecture of multimodal cnn.

from the three types of sensory data separately. The weights are not shared across the three branches. We apply a 3-layer CNN on each individual branch. In each layer, we employ 2D convolution filters with size of 5×3 to distill information across time and axes. Pooling is applied on the first two layers to reduce the dimensionality along time. Afterwards, we concatenate the outputs of the three individual networks along the channel dimension and feed them to the merged convolutional layer, where we also adopt 2D convolution of 3×3 to extract cross-modality features. Finally, We then flatten the outputs of the merged layers and feed them into the fully-connected layers to get the vector of this clip.

For all the layers, we use ReLU as the activation function. We also apply batch normalization (BN) at each convolutional layer. BN guarantees the input distribution of each layer remains unchanged across different mini-batches, leading to faster training speed and better generalization. Besides, we use dropout after the fully connected layer to prevent overfitting.

C. Multi-task Encoder-decoder Network

In this paper, we aim to achieve end-to-end word-level handwriting recognition without letter segmentation. To be more formal, given a word-level signal clip sequence $X = [x_1, x_2, \cdots, x_N]$ and the corresponding word $Y = [y_1, y_2, \cdots, y_U]$, where y_i represents a letter. We want to build networks to accurately map X to Y which will maximize the probability $P(Y|X)$, but in the training phase, we do not know which letter label in Y corresponds to which clips in X.

As far as we know, there is no prior work that exploits motion signals to enable end-to-end handwriting recognition. Motivated by the sequence learning networks in speech recognition and Neural Machine Translation, we design our network based on the attention-based encoder-decoder network [13].

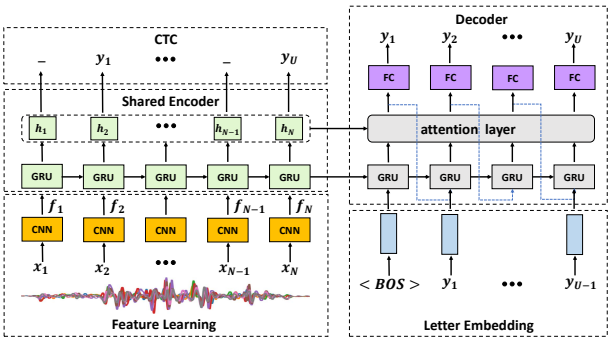

Fig. 8. The architecture of multi-task encoder-decoder network.

The attention network can automatically learn the language model in label sequences. This kind of ability is very important for handwriting recognition considering that the motion patterns of some letters may be indistinguishable in free-style handwriting, in which scenario the language model can boost the recognizing accuracy at word-level. However, the attention mechanism is purely data driven and thus too flexible on alignment prediction [20]. Especially when the input signals are noisy, it is difficult to find the best alignment between inputs and outputs [21]. To remedy this, we exploit another end-to-end method, i.e., Connectionist Temporal Classification(CTC), to guide the alignment prediction in the training of the attention model. Although CTC is often claimed to perform worse than the attention model in speech recognition [13], it enforces monotonic left-to-right alignment between input signals and output labels, which can speed up the process of estimating the desired alignment. Therefore, we combine CTC and attention into a multi-task sequence-to-sequence learning framework which absorbs the merits of them while counterbalancing their defects mutually. The network architecture is depicted in Fig. 8.

Encoder. In the encoder phase, the output of CNN ($F = [f_1, f_2, \cdots, f_N]$) is fed into a three-layer bidirectional GRU (Gated Recurrent Units) [22] to model temporal changes and output the hidden state as:

$$o_n = Encoder(f_n, o_{n-1}) \tag{5}$$

where o_n is the hidden state produced by GRU unit n. The reason why we use GRU rather than LSTM (Long short-term memory) is the two are very similar in modeling and performance but GRU has fewer parameters (two vs three gates), making it more suitable for our light-weight handwriting dataset. The final output vector o_N of GRU will serve as the latent vector which is passed to the decoder. We also save all the output sequence $O = [o_1, o_2, \cdots, o_N]$ for the attention mechanism and CTC.

Attention-based Decoder. The decoder is also a RNN network. The key idea of the attention mechanism is to assign a weight α_u^n for each encoder output o_n at each step u of the decoder and generate a context vector c_u. α_u^n can be interpreted as the contribution of an encoder output o_n to generate the word y_u. The inner working of the attention mechanisms is described in [20]. The hidden state h_u of the decoder at step u can be updated as:

$$y_u, h_u = Decoder(h_{u-1}, c_u, g_{u-1}) \tag{6}$$

y_u is the predicted letter label. g_{u-1} is the letter embedding of y_{u-1}. The initial h_0 is the latent vector o_N, and y_0 is a special label $< BOS >$ indicating the start of a word. The decoder phase will end when a label $< EOS >$, which indicates the end of the sequence, is predicted.

In the training phase, the probability of label y_u at step u can be calculated based on the above equations, so the conditional probability of the target word Y given the signal stream X is:

$$p(Y|X) = \prod_u p(y_u) \tag{7}$$

Therefore, we can minimize the corresponding cross entropy loss of $\mathcal{L}_{attention}$ to update all of the network parameters. Compared to CTC, the attention-based encoder-decoder network infers the current label by explicitly using the history label information, as depicted in Equation 6. In addition, its alignment is soft alignment which means the decoder can focus on multiple encoder outputs at each step. However, the attention alignment does not have the left-to-right monotonic constraint as it allows the decoder to attend anywhere in the output sequence at each step. In our handwriting recognition, user diversity and free-style handwriting will distort the motion signals significantly, which is very likely to generate misalignments by purely data-driven attention mechanism.

Multi-task Learning. In WriteAS, we use the CTC objective function to guide the training of the attention mechanism. As shown in Fig. 8, CTC and attention share the same encoder network. In CTC, the outputs of the encoder will pass through a softmax layer to get the possibility distribution for each clip. CTC will first find all the frame-wise alignments between the input feature clips F and the output labels Y. As the length of F is usually much larger than that of Y, CTC (a) introduces a special blank label '$-$' which corresponds to clips without true labels and (b) allows repetition of labels for both blank and non-blank labels. Thus, there can be different alignments corresponding to the same label sequences (e.g., both (a,-,b,b, c) and (a, a, b,c,c) correspond to (a, b, c)). Note that the alignments in CTC, although varied, own the same left-to-right label order with the label sequences. Given the probability distribution for each clip, CTC computes the probability of the target label sequence by marginalizing over all corresponding alignments. The CTC loss is also the cross entropy of this probability. We construct the overall objective function of WriteAS by combining both CTC and attention as follows:

$$\mathcal{L}_{word} = (1-\lambda)\mathcal{L}_{attention} + \lambda\mathcal{L}_{CTC} \tag{8}$$

with a tunable parameter λ ($0 \leqslant \lambda < 1$), which trades off the impact of attention and CTC. The network is trained by standard forward-backward algorithm. In this way, the left-to-right constraint of the CTC alignment will enforce the attention

mechanism to generate a monotonic alignment between the signals and the output label sequences, which can speed up the alignment searching as well as enhance the robustness to signals distortions.

Our encoder network is a 2-layer bidirectional GRU with 128 cells in each layer while the decoder network is a 2-layer GRU with 128 cells. We use letter as the basic modeling unit. The letter, before fed into the decoder, needs to be represented by a vector. We use the word embedding [23] to encode the letters. Specifically, we use a fully connected layer that learns a linear projection from one-hot vectors of letters to a denser space. In the training phase, we use Adam optimizer [24] to minimize the corresponding cross entropy loss of L_{word}. In the decoder phase, given the posterior probability distributions of labels at each step, we employ the beam search algorithm [25] to generate the final sequence.

IV. Evaluation

In this section, we first introduce the experimental setup, including the dataset collection and the evaluation protocol. Then, we evaluate the performance of WriteAS under different conditions to validate its effectiveness. and robustness.

A. Experimental Setup

Data Collection. We implement a data collection prototype on a TicWatch E2 smartwatch with an Android system. The sampling rate of the inertial sensors is set to 100Hz. We design a word-level dataset which consists of 240 commonly used English Words. All the 26 English characters are included in these words. The length of each word ranges from 1 (characters) to 9 and the average length is 4.32. 12 volunteers (9 males and 3 females, aged from 20 to 30) are recruited to participate in the data collection. Before the experiments, we inform them the objective and procedure of this study. Volunteers wear the smartwatch in their dominant hands (the right hand) and write on the desk (the surface material is wood). They write the words in their habitual ways. To meet the real usage scenario, there are no restrictions on volunteers' writing speed, strength and habits. The locations that volunteers perform the experiments are different from each other, which mainly contain four settings, i.e. laboratory, meeting room, bedroom and living room. Each volunteer performs 2 sessions and repeats each words for 5 times at each session. Different sessions are in different time to ensure the data diversity. The whole collection process lasted for one months. The dataset contains a total of 28800 samples. We will publish the code and dataset to the github in the future.

Evaluation Protocol. We evaluate our method by three evaluation strategies: **(1) user-dependent test:** User-dependent means that each user appears in both the training and testing set. **(2) user-independent test:** We perform leave-one-user-out cross-validation to validate the capacity of our method to deal with user diversity. **(3) unseen word test:** We also evaluate the performance in translating unseen word (word not in the training set). As there is no public and large-scale dataset for smartwatch-based handwriting recognition, the recognition ability for unseen words can eliminate the burden to collect all possible words. During the inference of the network, we do not restrict the length or the letter order of the word and thus there is an enormous amount of possible label sequences. This makes the word-level recognition, especially for unseen word recognition, more challenging.

We employ character error rate (CER) as the criterion, which is widely used in the scope of continuous speech recognition. CER measures the least operations of substitution, deletion and insertion to transform a hypothesized word to the ground truth:

$$CER = \frac{N_{sub} + N_{del} + N_{ins}}{N_{ground\ truth\ words}} \tag{9}$$

Where N_{sub}, N_{del}, and N_{ins} denote the number of required substitutions, deletions and insertions, respectively.

In the training phase, we perform data augmentation for each samples with rate $= 20\times$, meaning that the number of samples increases 20 times compared with the original one. We use Adam optimizer [24] with a default setting. The model is trained in a server with Intel Core i9-7900X CPU @ 3.3GHz, 64GB memory and 4 Nvidia GTX 1080 Ti GPU cards.

B. Overall Performance

User-dependent test. In the user-dependent test, we adopt k-fold cross validation to evaluate the recognition performance. Specifically, the sample set is randomly split into k smaller sets, where $k = 5$ in our implementation. A model is trained using $k-1$ of the folds and validated on the remaining part of the data. WriteAS can achieve a CER of 0.6%. We do not use data augmentation here due to the high recognition performance. The average word length in our dataset is 4.32, so 0.6% means that only a letter needs to be corrected every 40 words. The main reason why the system performs so well is that users in the testing set also appear in the training set, leading to similar signal distribution between the training data and the testing data.

User-independent test. Fig. 9 is the result across volunteers under leave-one-user-out test. The average CER is 8.1%. We can see that everyone's CER is lower than 12% (ranging from 4.7% to 11.8%). The result is very impressive considering that (a) the user is not in the training set; (b) no external language model is used to improve the performance. Fig. 10 presents the CER over different word lengths. We can see that the CER exhibits a clear increasing trend with the length decreasing in user-independent test. The CER is 12.3% when the word length is 1 (letter) and decreases to 5.4% when there are nine letters in the word. This is reasonable because our decoder network can make a context-based inference via the self-learned language model and longer words means that more context information can be captured.

Unseen word test. We further conduct evaluations on recognizing unseen words. In each test, we select the data of 20 words as testing set and the data of the remaining words as training set. The average CER is 9.7%. The excellent performance eliminates the burden of collecting all the English

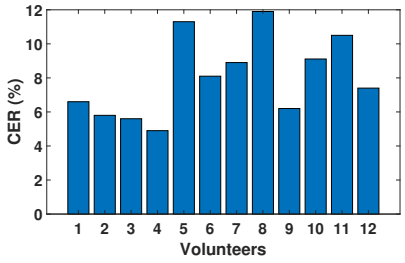

Fig. 9. CER across users.

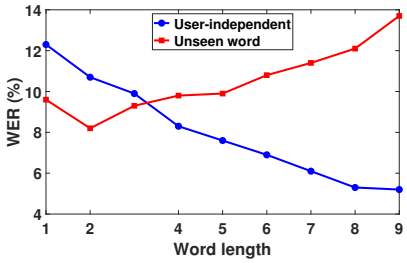

Fig. 10. CER over different word lengths.

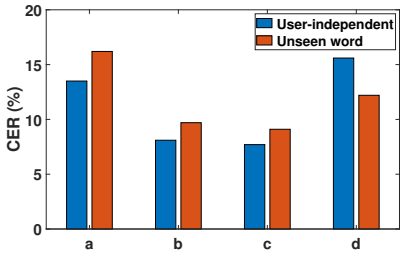

Fig. 11. CER of different inertial signals.

words. The CER over different word lengths in this test, as shown in Fig. 10 shows the reversing trend with that in user-dependent test. The CER ranges from 8.2% to 14.7% . When the word is not in the training set, the language model learned by the attention mechanism helps little for the inference, so longer words tend to bring more errors. With the number of the words in the training data increasing, the language model will play a bigger role and we will discuss this in the following section.

C. Micro-benchmark

We evaluate the micro-benchmark in this subsection. As the accuracy of user-dependent test is too high which may not manifest the effect of these parameters, we perform the following evaluations under user-independent test and unseen word test, respectively.

Inertial Signals. Our system leverage the signals from multiple inertial sensors, i.e., accelerometer, gyroscope and gravity, to capture the hand movements. The accelerometer refers to the linear accelerometer which has removed the gravity from the raw acceleration. In this part, we investigate whether it is redundant or insufficient to use signals from these sensors. We perform evaluations in three settings, i.e., *(a) Linear accelerometer + gyroscope, (b) Linear accelerometer + gyroscope + gravity and (c) Linear accelerometer + gyroscope + gravity + raw accelerometer.* We also evaluate the impact of data augmentation on our system. Fig. 11 shows the results in the four settings. *d* represents the result without data augmentation. For one thing, we can see that the gravity data can improve the system performance notably while the improvement when using 4 sensors is little. For another thing, we can see a clear improvement when using data augmentation, especially in the user-independent test. This is because our data augmentation mechanism can cover users' different writing behaviors (speed, font size and watch position) and thus enhance the generalization capability of the system.

CNN front-end. In WriteAS, we use a multimodal CNN as the front-end to abstract features from multiple sensors. In this part, we investigate the impact of different CNN front-ends. We perform evaluation by replacing the multimodal CNN with the following CNN models:

- **naive CNN (NC):** This is a 2D CNN model without individual branch of modalities (5 layers).
- **DeepFusion (DF) [26]:** This model also has the individual CNN for each modality. Compared with our network, it introduces the attention mechanism to the multi-sensor fusion. It uses a small fully connected neural network to learn the weight of sensors directly based on their inputs.
- **GlobalFusion (GF) [27]:** This model applies a more elaborate attention mechanism. It uses the features output by the merged CNN as the global query to estimate the weight of each modality.

Fig. 12 is the results of different CNN front-ends. The naive CNN gets the highest CER (user-independent: 17.1%, unseen word: 20.5%), indicating the importance of the multimodal fusion. Both DeepFusion and GloablFusion dynamically adjust the weight of each modality based on their inputs while the weights of our method is fixed after training. Theoretically the two methods can yield better performance, but it can be seen from the figure that they improve little or even decrease. This is because their modality-attention mechanisms are more suitable for the whole-body or whole-arm activities, in which the representation capability of each sensor vary notably with different activities. For example, the accelerometer features may be more significant in distinguishing the 'walking' and 'biking' activities while the gyroscope features may be more significant in distinguishing the 'turning-left' and 'turning-right' activities [28]. When it comes to handwriting recognition, nearly all the variations in writing behaviors are at the micro level. As a result, the representation capability of each sensor changes little over different letters and words.

End-to-end networks. In this part, we make a comparison with CTC and attention network. The CTC network is similar with Fig. 8 but has a 3-layer bidirectional GRU. The attention network is just the same with our multi-task network with $\lambda = 0$. Results in Fig. 13 show that our system exhibits clearly better performance than the other two end-to-end networks. In the user-independent test, the performance of the attention network is better than that of CTC due to its capability in learning the language model. In contrast, CTC performs much better in the unseen word test. As mentioned in Section IV-B, the language model learned by the attention network does little for unseen word recognition. Worse still, it is likely to generate an error word that is in the training set due to misalignments. CTC does not suffer from this and can generate alignments more quickly and accurately due to its left-to-right constraints. Our multi-task framework combines the advantages of these

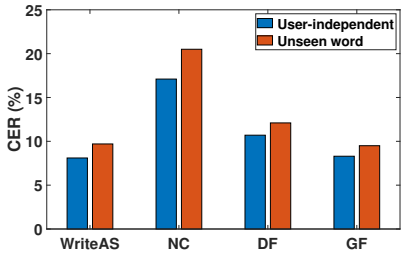

Fig. 12. Comparison with other CNN front-end.

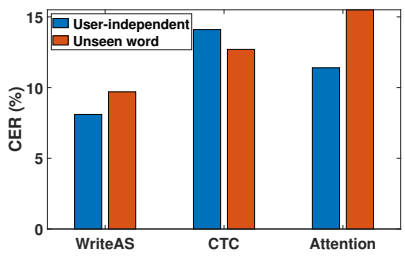

Fig. 13. Comparison with other end-to-end networks.

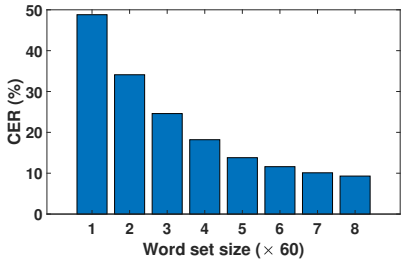

Fig. 14. CER over dictionary size.

TABLE I
CER UNDER DIFFERENT WRITING SPEED.

Speed	CER
Fast	0.082
Normal	0.066
Slow	0.071

TABLE II
CER UNDER DIFFERENT WRITING SURFACE MATERIALS.

materials	CER	materials	CER
Wood	0.075	iron	0.081
plastic	0.088	marble	0.069
glass	0.086	cardboard	0.094

TABLE III
CER UNDER DIFFERENT SMARTWATCH MODELS.

Models	CER
Ticwatch	0.086
HuaWei	0.138
Samsung	0.129

two networks and thus yields the best performance.

D. System Robustness.

In this subsection, we investigate the robustness of our system to different impact factors, including writing speed, surface and smartwatch model. We perform user-independent test for each evaluation.

Word set size. As we do not use any external language model, the word set size of the training data is important to the performance of our system, especially for the unseen word recognition. Due to labor and time costs, our word-level handwriting dataset only includes 240 words, which are a small part of the commonly used English words. With more words, the encoder-decoder can learn better language model. In this part, we investigate the impact of the word set size. we ask 2 volunteers to write another 260 words and 10 times for each word. Then we extract the data of them from the original dataset and construct a new dataset of 500 words. We select 20 words for unseen word test, and the remaining are used as training data. We vary the word set size of the training data from 60 to 480 and perform evaluations. The results are shown in Fig. 14. We can see that the CER decreases significantly with the word set size increasing. Although the trend tends to be smooth when exceeding 240, it still decreases gradually. The unseen word test in Section IV-B shows that WriteAS can achieve a CER of 9.7% for unseen words. We think this accuracy is sufficient for daily life use. But we can further improve the performance by collecting the data of more words.

Writing speed. In this part, we investigate the impact of writing speed on our system. In our dataset, the average writing speed is about 4 letters per 3 seconds. We use this speed as the normal speed and ask two volunteers to write 120 words for 2 times in fast, normal, and slow speeds: the fast speed is to write three letters per two seconds and the slow speed is to write one letter per second. The results are displayed in Table I, which show that the performance will not experience significant degradation for both fast and slow speed.

Surface for writing. In this part, we investigate the impact of surface materials on our system. We also ask two volunteers to write 120 words for 2 times in different materials, including the wood table, iron pad, plastic pad, marble, glass and cardboard. The results in Table II show that these materials have very little impact on the performance of WriteAS. This is a remarkable advantage compared with those acoustic-based methods, which degrade severely on some surfaces [8]. The reason is that their performance is highly dependent on the sound of the friction which may differ significantly from material to material.

Smartwatch models In this part, we investigate the impact of different smartwatches on our system. We use three Android smartwatches, i.e., Ticwatch E2 in this paper, HuaWei Honor 2 and Samsung Gear S2, which differ in size, sensor position and data quality. The sampling rate is set to 100KHz. We ask four volunteers to write all the 240 words for 5 times using the other two watches. Table III shows the results. We can see that the performance degrades on the other two watches. However, the results are still acceptable considering that the all the training data is collected using Ticwatch. When we add the data from one of the other two watches into the training data, the CER can decrease notably (with CER lower than 10%).

E. System Delay

To evaluate the real-time performance of WriteAS, we implement an app on the watch . Meanwhile, we use a laptop (Intel Core i7-7700HQ CPU@2.8GHz and 16GB RAM) as

the cloud server. The app is only used for data collection and result display. The deep learning module are implemented on the server. The system delay is defined as the time difference between the user ending the handwriting and WriteAS displaying the result on the screen of the watch. We ask a volunteer to write 60 words for one time and record the system delay for each writing. The system delay mainly consists of three parts, i.e., data uploading, word segmentation and forward feeding of deep network. As the data will upload once the user starts the app, the network delay can be small ($< 20ms$). The word segmentation module will bring some latency because we use a sliding window of 100ms (50% overlap) to perform writing detection. The inference time of the deep learning network for a word of 9 letters is 86ms. The average system delay for different words is 166ms. As users usually pause between two adjacent words for more than 500ms [9], they are not clearly aware of such a response time, indicating that WriteAS can recognize the word in real time. In WriteAS, the smartwatch is only used for collecting data and display results, so we do not present the power consumption in this paper as too much previous work has done it [29] [30].

F. Compare with touch input method.

In this subsection, we compare WriteAS with touch input method. We ask 10 volunteers to input 100 words using WriteAS and touch input, respectively. 5 of them (Group 1) have participated our data collection experiments while the remaining people (Group 2) are the first time to use our system, which means the recognition for their writing is user-independent. Before the evaluation, we ask the 5 new users to learn our system for one minute. Each time the smartwatch outputs a wrong letter or word, the users should correct it. If WriteAS gives the error results for three times, users will turn to use touch input to type this letter. The average input speeds of WriteAS for Group 1 and Group 2 are 15.4 WPM (words-per-minute)and 11.7 WPM, respectively. In contrast, the average speed of the touch input method is only 8.1 WPM. The average CER of WriteAS for Group 1 and Group 2 are 2.4% and 10.2% while that of touch input is 17.4%. When using touch input, volunteers waste much time on aiming at the tiny keypads of the watch screen and correcting their erroneous. In contrast, our system provide a better input experience for users. The evaluation show that our system outperforms touch input by a large margin in both accuracy and efficiency.

V. Related Work

Acoustic-based handwriting recognition. The acoustic-based handwriting recognition systems can be grouped into two categories, i.e., passive sensing and active sensing. The passive sensing methods leverage the sound generated by handwriting on the table for recognition. WritingHacker [5] makes the first exploration on this field, but it only achieves 50% 60% accuracy. WordRecorder [6] designs a deep neural network, which extracts the depth information to build a user-independent model. Both of them can only recognize print-style capital letters. But in daily life, lowercase letters are much more required. To solve this problem, WritingRecoder builds a model to enable free-style lowercase handwriting recognition. It uses the Short-time power spectral density as features and models the time-series relations by a CNN+LSTM framework. These methods are vulnerable to environmental noises as well as surface materials. The active sensing methods actively send the inaudible acoustic signal and capture the hand movements according to the reflected signal [31] [32]. EchoWrite [33] decomposes the capital letters into six basic strokes and enable training-free handwriting recognition. The major drawback of passive sensing methods is that their performance is highly position-dependent, and the systems are likely to fail with subtle changes in writing positions.

Sensor-based handwriting recognition. The inertial sensors in mobile devices have enabled a broad range of HCI applications [34] [35] [29], which also make it possible for in-air or on-surface handwriting recognition . ArmTrak [36] employs the inertial sensors on a smartwatch to track the 3D posture of the entire arm. AirContour [10] leverages the arm contour in 3D space to enable in-air writing. They need the movements of the whole-arm which tend to make users exhausted. SHOW [9] achieves the state-of-the-art handwriting recognition performance on smartwatch. It uses the elbow as the support point to recognize individual letters. In addition, it adopts a word selection method based on language model to recognize legislate words from all possible letter combinations. Compared to SHOW, our method can enable word-level recognition in an end-to-end way which gets around letter segmentation. This allows users to write in a cursive and continuous way without pausing between adjacent letters. There are other researches use the sensors embedded in a pen to enable interaction with mobile devices. For example, MagHacker [11] recognizes letters by monitoring the magnetic field being produced by the stylus pen's internal magnet. Pentelligence [12] combines inertial sensors and microphones together into a pen to recognize handwritten digits. However, these external devices are not suitable for text input on smartwatch.

VI. Conclusion

In this paper, we present WriteAS, a first-of-its-kind text input system on smartwatch which enables end-to-end handwriting recognition at word-level. The key techniques are a multimodal CNN front-end which aggregates information across multiple modalities, and a multi-task encoder-to-decoder network which delivers word-level handwriting recognition without letter segmentation. We construct a high-quality dataset and achieve impressive performance under a broad range of evaluations. Specifically, the CER of our system are 0.6% and 8.1% in user-dependent and user-independent settings, respectively. Besides, our system can achieve a CER of 9.7% on the recognition of unseen words. These results demonstrate the considerable potential of WriteAS in handwriting recognition filed as well as provide a series of strong baselines for the research community.

REFERENCES

[1] H. Gil, H. Kim, and I. Oakley, "Fingers and angles: Exploring the comfort of touch input on smartwatches," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 2, no. 4, Dec. 2018.

[2] B. Lafreniere, C. Gutwin, A. Cockburn, and T. Grossman, "Faster command selection on touchscreen watches," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ser. CHI '16, 2016, p. 4663–4674.

[3] A. Gupta and R. Balakrishnan, "Dualkey: Miniature screen text entry via finger identification," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ser. CHI '16, 2016, p. 59–70.

[4] J. Rico and S. Brewster, "Gesture and voice prototyping for early evaluations of social acceptability in multimodal interfaces," in *International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction*, ser. ICMI-MLMI '10. Association for Computing Machinery, 2010.

[5] T. Yu, H. Jin, and K. Nahrstedt, "Mobile devices based eavesdropping of handwriting," *IEEE Transactions on Mobile Computing*, vol. 19, no. 7, pp. 1649–1663, 2020.

[6] H. Du, P. Li, H. Zhou, W. Gong, G. Luo, and P. Yang, "Wordrecorder: Accurate acoustic-based handwriting recognition using deep learning," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, 2018, pp. 1448–1456.

[7] M. Chen, P. Yang, J. Xiong, M. Zhang, Y. Lee, C. Xiang, and C. Tian, "Your table can be an input panel: Acoustic-based device-free interaction recognition," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 3, no. 1, Mar. 2019. [Online]. Available: https://doi.org/10.1145/3314390

[8] H. Yin, A. Zhou, G. Su, B. Chen, L. Liu, and H. Ma, "Learning to recognize handwriting input with acoustic features," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 4, no. 2, Jun. 2020.

[9] X. Lin, Y. Chen, X.-W. Chang, X. Liu, and X. Wang, "Show: Smart handwriting on watches," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 1, no. 4, Jan. 2018.

[10] Y. Yin, L. Xie, T. Gu, Y. Lu, and S. Lu, "Aircontour: Building contour-based model for in-air writing gesture recognition," *ACM Trans. Sen. Netw.*, vol. 15, no. 4, Oct. 2019. [Online]. Available: https://doi.org/10.1145/3343855

[11] Y. Liu, K. Huang, X. Song, B. Yang, and W. Gao, "Maghacker: Eavesdropping on stylus pen writing via magnetic sensing from commodity mobile devices," in *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 148–160. [Online]. Available: https://doi.org/10.1145/3386901.3389030

[12] M. Schrapel, M.-L. Stadler, and M. Rohs, "Pentelligence: Combining pen tip motion and writing sounds for handwritten digit recognition," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, ser. CHI '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1–11. [Online]. Available: https://doi.org/10.1145/3173574.3173705

[13] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[14] A. Graves, S. Fernández, and F. Gomez, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *International Conference on Machine Learning*, 2006.

[15] M. T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," 2015.

[16] C. Shen, B. Ho, and M. Srivastava, "Milift: Efficient smartwatch-based workout tracking using automatic segmentation," *IEEE Transactions on Mobile Computing*, vol. 17, no. 7, pp. 1609–1622, 2018.

[17] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 2613–2617. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-2680

[18] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, 2011, pp. 689–696.

[19] K. Sohn, W. Shang, and H. Lee, "Improved multimodal deep learning with variation of information," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 2141–2149.

[20] Y. B. Dzmitry Bahdanau, Kyunghyun Cho, "Neural machine translation by jointly learning to align and translate," 2015.

[21] Y. Pan, T. Mei, T. Yao, H. Li, and Y. Rui, "Jointly modeling embedding and translation to bridge video and language," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[22] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv: Neural and Evolutionary Computing*, 2014.

[23] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv: Learning*, 2014.

[25] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[26] H. Xue, W. Jiang, C. Miao, Y. Yuan, F. Ma, X. Ma, Y. Wang, S. Yao, W. Xu, A. Zhang, and L. Su, "Deepfusion: A deep learning framework for the fusion of heterogeneous sensory data," in *Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, ser. Mobihoc '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 151–160. [Online]. Available: https://doi.org/10.1145/3323679.3326513

[27] S. Liu, S. Yao, J. Li, D. Liu, T. Wang, H. Shao, and T. Abdelzaher, "Giobalfusion: A global attentional deep learning framework for multisensor information fusion," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 4, no. 1, Mar. 2020. [Online]. Available: https://doi.org/10.1145/3380999

[28] H. Ma, W. Li, X. Zhang, S. Gao, and S. Lu, "Attnsense: Multi-level attention mechanism for multimodal human activity recognition," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 7 2019, pp. 3109–3115. [Online]. Available: https://doi.org/10.24963/ijcai.2019/431

[29] J. Hou, X.-Y. Li, P. Zhu, Z. Wang, Y. Wang, J. Qian, and P. Yang, "Signspeaker: A real-time, high-precision smartwatch-based sign language translator," in *The 25th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '19. New York, NY, USA: Association for Computing Machinery, 2019. [Online]. Available: https://doi.org/10.1145/3300061.3300117

[30] L. Jiang, X. Lin, X. Liu, C. Bi, and G. Xing, "Safedrive: Detecting distracted driving behaviors using wrist-worn devices," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 1, no. 4, Jan. 2018. [Online]. Available: https://doi.org/10.1145/3161179

[31] W. Wang, A. X. Liu, and K. Sun, "Device-free gesture tracking using acoustic signals," in *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, 2016, pp. 82–94.

[32] S. Yun, Y. Chen, H. Zheng, L. Qiu, and W. Mao, "Strata: Fine-grained acoustic-based device-free tracking," pp. 15–28, 2017.

[33] Y. Zou, Q. Yang, R. Ruby, Y. Han, S. Wu, M. Li, and K. Wu, "Echowrite: An acoustic-based finger input system without training," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 2019, pp. 778–787.

[34] A. Parate, M.-C. Chiu, C. Chadowitz, D. Ganesan, and E. Kalogerakis, "Risq: Recognizing smoking gestures with inertial sensors on a wristband," in *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 149–161. [Online]. Available: https://doi.org/10.1145/2594368.2594379

[35] T. H. Vu, A. Misra, Q. Roy, K. C. T. Wei, and Y. Lee, "Smartwatch-based early gesture detection 8 trajectory tracking for interactive gesture-driven applications," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 2, no. 1, Mar. 2018. [Online]. Available: https://doi.org/10.1145/3191771

[36] S. Shen, H. Wang, and R. Roy Choudhury, "I am a smartwatch and i can track my user's arm," in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 85–96. [Online]. Available: https://doi.org/10.1145/2906388.2906407