SignFi: Sign Language Recognition Using WiFi

YONGSEN MA, GANG ZHOU, SHUANGQUAN WANG, HONGYANG ZHAO, and WOOSUB JUNG, Computer Science Department, College of William and Mary, USA

We propose SignFi to recognize sign language gestures using WiFi. SignFi uses Channel State Information (CSI) measured by WiFi packets as the input and a Convolutional Neural Network (CNN) as the classification algorithm. Existing WiFi-based sign gesture recognition technologies are tested on no more than 25 gestures that only involve hand and/or finger gestures. SignFi is able to recognize 276 sign gestures, which involve the head, arm, hand, and finger gestures, with high accuracy. SignFi collects CSI measurements to capture wireless signal characteristics of sign gestures. Raw CSI measurements are pre-processed to remove noises and recover CSI changes over sub-carriers and sampling time. Pre-processed CSI measurements are fed to a 9-layer CNN for sign gesture classification. We collect CSI traces and evaluate SignFi in the lab and home environments. There are 8,280 gesture instances, 5,520 from the lab and 2,760 from the home, for 276 sign gestures in total. For 5-fold cross validation using CSI traces of one user, the average recognition accuracy of SignFi is 98.01%, 98.91%, and 94.81% for the lab, home, and lab+home environment, respectively. We also run tests using CSI traces from 5 different users in the lab environment. The average recognition accuracy of SignFi is 86.66% for 7,500 instances of 150 sign gestures performed by 5 different users.

CCS Concepts: • **Human-centered computing** → **Gestural input**; **Interaction design**;

Additional Key Words and Phrases: Sign Language Recognition, Gesture Recognition, Wireless, Channel State Information, Convolutional Neural Network.

ACM Reference Format:
Yongsen Ma, Gang Zhou, Shuangquan Wang, Hongyang Zhao, and Woosub Jung. 2018. SignFi: Sign Language Recognition Using WiFi. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 1, Article 23 (March 2018), 21 pages. https://doi.org/10.1145/3191755

1 INTRODUCTION

According to the World Federation of the Deaf (WFD), there are 70 million deaf people using sign language as their first language; many hearing people also use sign language as their first or second language[1]. In the U.S. alone, there are one half to two million people using American Sign Language (ASL) in the 1990s [17]. Many colleges accept ASL as a foreign language credit, and more people are learning and using ASL. Modern Language Association conducted a survey of course enrollments in languages other than English from 2,696 institutions in the U.S. [9]. According to the survey, the number of ASL enrollments is consistently increasing from year 2002 to 2013. There are 109,577 ASL enrollments at 2013, making ASL the language with the third most enrollments.

[1]https://wfdeaf.org/human-rights/crpd/sign-language/

Authors' address: Yongsen Ma, yma@cs.wm.edu; Gang Zhou, gzhou@cs.wm.edu; Shuangquan Wang, swang10@email.wm.edu; Hongyang Zhao, hzhao03@email.wm.edu; Woosub Jung, wjung01@email.wm.edu, Computer Science Department, College of William and Mary, 251 Jamestown Rd. Williamsburg, VA, 23187-8795, USA.

2474-9567/2018/3-ART23 $15.00
https://doi.org/10.1145/3191755

There is a huge barrier between the Deaf community and people that do not understand or know little about sign language. A sign language recognition system would help break this barrier. There are some sign language recognition systems using cameras [34] or Kinect [13, 28, 36, 47], but they are subject to lighting conditions. Some systems use Leap Motion [6–8, 20, 23, 24, 30, 36, 47], but they can recognize only finger gestures and are very sensitive to the distance and displacement of the Leap Motion sensor and the human. Some systems use gloves and motion sensors, like SignAloud [26], but they are intrusive and require sensors to be attached on fingers.

Many papers have shown that Channel State Information (CSI) of WiFi can be used to recognize hand [1, 21, 29, 33, 37, 40] and finger [18, 21, 38, 48] gestures in a non-intrusive way. WiFi signals are used to recognize ASL gestures in [18, 21, 33]. These are the most relevant papers to our work. But they are only evaluated on simple ASL gestures: 5 hand gestures in [33], 9 finger postures in [18], and 25 hand/finger gestures in [21]. Our object is to recognize nearly 300 basic sign gestures [35] that are frequently used in daily life. *The classification algorithm should have high recognition accuracy and low computational cost during testing.*

Existing classification algorithms have very low recognition accuracy when the number of sign gestures increases to nearly 300. Both papers [18, 21] use k-Nearest Neighbor (kNN) with Dynamic Time Wrapping (DTW) as the classification algorithm. We test it in a lab environment using CSI traces of 276 sign gestures. The average recognition accuracy of kNN with DTW is only 68% for 276 sign gestures. Moreover, kNN with DTW takes extremely long time in the testing stage when there are nearly 300 possible classes. Thus, new classification algorithms are needed for sign gesture recognition using WiFi.

We propose SignFi to accurately recognize sign gestures using a 9-layer Convolutional Neural Network (CNN). It collects CSI measurements to capture wireless signal characteristics of sign gestures. After removing noises, SignFi feeds the pre-processed CSI measurements to a 9-layer CNN for sign gesture classification. We collect CSI traces for 276 sign gestures, each with 20 instances for the lab environment and 10 instances for the home environment. The average recognition accuracy of SignFi is 98.01%, 98.91%, and 94.81% for the lab, home, and lab+home environment, respectively. Fig. 1 compares SignFi with existing sign language recognition technologies. Most of the existing technologies are tested on simple ASL gestures, such as 9 digital numbers and 26 alphabet letters. SignFi is the only one that is able to recognize 276 sign gestures with 94.81% accuracy. In summary, we make the following contributions:

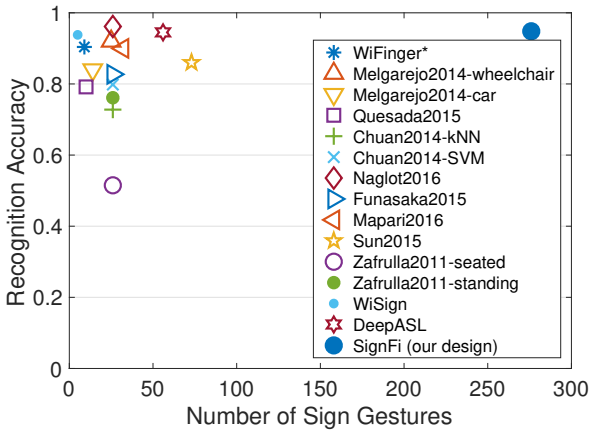


Fig. 1. Comparison of different sign language recognition technologies. More details are shown in Table 4.

(1) We propose a new signal processing technique to remove noises from raw CSI measurements. The information about how CSI changes over sub-carriers and sampling time is recovered.
(2) We present a 9-layer Convolutional Neural Network for accurate sign gesture recognition using WiFi signals.
(3) Our design has above 94% accuracy for 8,280 instances of 276 sign gestures from lab and home environments. We also run tests on 7,500 instances of 150 sign gestures from 5 different users and get 86.66% accuracy.

The rest of the paper is organized as follows. Section 2 gives background and motivation of sign language recognition using WiFi signals. The SignFi design, including signal processing and a 9-layer CNN, is presented in Section 3. Section 4 shows experiment setup and evaluation results. Section 5 summaries related works and compares them with SignFi. Section 6 concludes the paper and discusses future work.

2 BACKGROUND AND MOTIVATION

Channel State Information (CSI) captures how wireless signals propagates from the transmitter to receiver at a certain carrier frequency. Wireless signals usually travel with multiple paths. For a multi-path channel with N paths, the CSI value for each transmit and receive antenna pair of each sub-carrier is

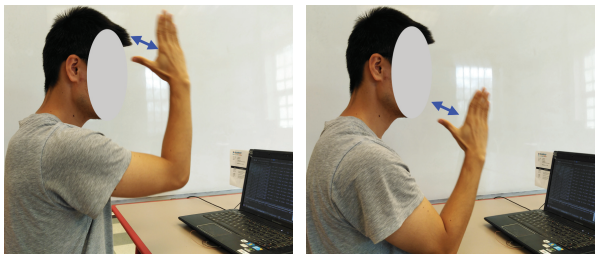
$$h = \sum_{n}^{N} a_n e^{-j2\pi d_n/\lambda + j\phi_n}, \tag{1}$$

where a_n is the attenuation factor, d_n is the path distance from the transmitter to the receiver, ϕ_n is the phase shift along the nth path, and λ is the carrier wavelength [39]. h is a complex number that can be expressed by its amplitude and phase. If either the amplitude or phase of at least one path changes, the CSI value changes.

Since CSI captures how wireless signals are reflected by surrounding objects, it can be used to recognize hand [1, 21, 29, 33, 37, 40] and finger [18, 21, 38, 48] gestures. Three papers [18, 21, 33] use WiFi signals to recognize ASL gestures: 5 hand gestures in [33], 9 digits finger postures in [18] and 25 hand/finger gestures in [21]. Our object is to recognize not only simple sign gestures but nearly 300 basic ASL gestures that are frequently used in daily life. *The classification algorithm should have high recognition accuracy and low computational cost during testing.* Both [18] and [21] use k-Nearest Neighbor (kNN) with Dynamic Time Wrapping (DTW) as the classification algorithm. The question is whether kNN with DTW still works when the number of sign gestures increases by one order of magnitude.

In terms of computational cost of testing, kNN with DTW is not very efficient when there are nearly 300 possible classes. Although kNN with DTW takes no time to train, it has extremely high overhead at testing time. For each testing sample, kNN with DTW needs to compare it with every single training sample. This requires a lot of computation resources during testing when there are 276 possible classes and each training/testing sample has 3,600 data points. Even for only 25 sign gestures, kNN with DTW needs more than 5 seconds per sign gesture during testing, which is shown later in Section 4.2.2.

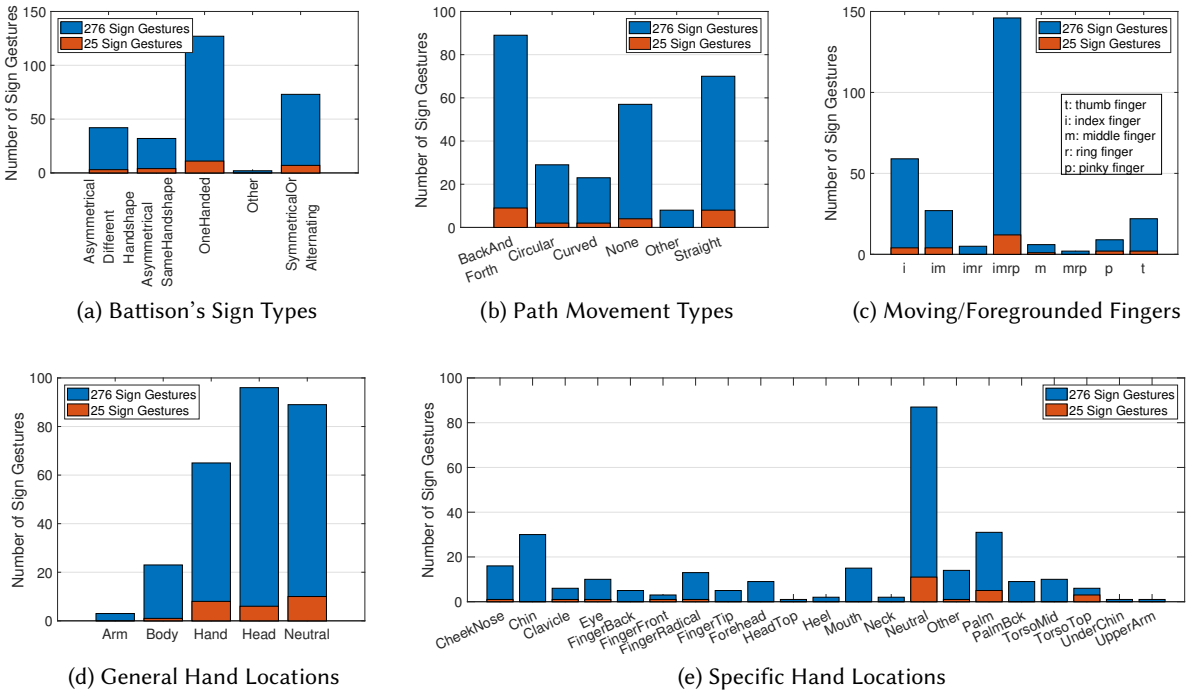
In terms of recognition accuracy, there are three challenges for recognizing nearly 300 sign gestures: (1) the number of sign gestures is large; (2) many different sign gestures have similar arm, hand, or finger movements; (3) many sign gestures involve complex and diverse movements. First, no more than 25 sign gestures are tested in [18, 21, 33] for sign gesture recognition using WiFi. There are nearly 300 basic sign words that are frequently used in daily life [35]. Second, since many sign words have similar gestures or postures, it is very hard to distinguish them from each other. For example, sign words "Father" and "Mother" have the same hand gesture and finger posture, but they require the dominant hand in different locations, as shown in Fig. 2. Finally, many sign gestures involve head, arm, hand, and finger movements. The dominant hand is not constrained in a small area; it can be near different parts of the human body.



(a) Sign gesture for "Father" (b) Sign gesture for "Mother"

Fig. 2. Sign words "Father" and "Mother" have the same hand gesture and finger posture, but they need the dominate hand in different locations.

For the 276 sign gestures used in our experiments, we check their movement types using the ASL-LEX database [2]. The database gives sign types, path movement types, general locations, specific locations, and moving/foregrounded fingers of the dominant hand for nearly 1,000 sign gestures. For the definitions and descriptions of each category, please check ASL-LEX [2]. We manually add the labels for gestures that are not included in ASL-LEX. Fig. 3 shows the number of sign gestures in each category. Many of the 25 sign gestures



(a) Battison's Sign Types (b) Path Movement Types (c) Moving/Foregrounded Fingers

(d) General Hand Locations (e) Specific Hand Locations

Fig. 3. Moving/foregrounded fingers, general and specific locations of the dominant hand of the 276 sign gestures used in our experiments. They involve more complex and diverse movements than the 25 sign gestures.

are evenly distributed in 3 or 4 categories, which makes them different from each other. For the 276 sign gestures, there are much more gestures in the same category, which makes them harder to be distinguished. For specific hand location, the 276 sign gestures have 12 categories that are not covered by the 25 sign gestures. Therefore, it is much harder to recognize the 276 sign gestures.

Can kNN with DTW still work for the 276 sign gestures? To answer this question, we collect CSI measurements and evaluate kNN with DTW in a lab environment. Fig. 4 shows the recognition accuracy of kNN with DTW for 25 and 276 sign gestures. For the 25 sign gestures, kNN with DTW has above 96% accuracy, which is in consistent with [18, 21]. However, the average accuracy drops to 68% for the 276 sign gestures. Thus, new algorithms are needed to improve recognition accuracy and reduce cost during testing for sign language recognition using WiFi signals. For this purpose, we propose SignFi using a 9-layer CNN as the classification algorithm. It has high recognition accuracy and low cost during testing.

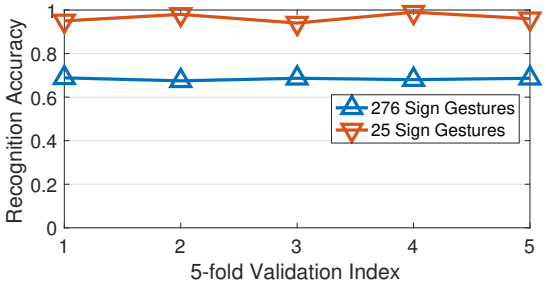


Fig. 4. The average recognition accuracy of kNN with DTW decreases from 96% to 68% when the number of sign gestures increases from 25 to 276.

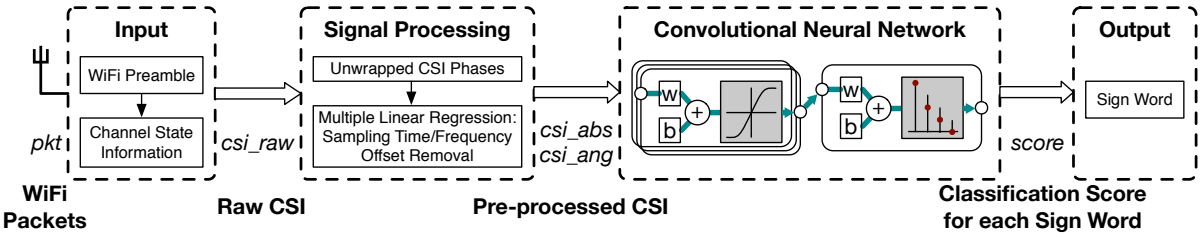


Fig. 5. SignFi Overview

3 SIGNFI DESIGN

The SignFi overview is shown in Fig. 5. SignFi collects CSI measurements by WiFi preambles. Raw CSI measurements are pre-processed to remove noises. Both amplitude and phase of the pre-processed CSI are fed to a 9-layer CNN for sign classification. In this paper, we mainly focus on the classification algorithm. CSI measurements are manually segmented for each sign gesture. We use these manually segmented CSI traces for both the proposed and existing classification algorithms for fair comparison. We leave automatic time segmentation to future work.

3.1 SignFi Signal Processing

We need to remove noises before feeding raw CSI measurements to the classification algorithm. In our experiments, we use a linear antenna array with 3 transmit antennas. In this case, the CSI of the ith antenna is

$$h_i = he^{-j2\pi(i-1)\Delta\cos\psi} = \left(\sum_n^N a_n e^{-j2\pi d_n/\lambda + j\phi_n}\right) e^{-j2\pi(i-1)\Delta\cos\psi}, \tag{2}$$

where h is the multi-path CSI (equation (1) in Section 2), Δ is the transmit antenna separation normalized to the unit of carrier wavelength, and ψ is the angle of departure with respect to the transmit antenna array [39]. CSI h_i captures the impact of multi-path channel propagation and the arrangement of transmit antenna array. The transmit antenna array adds the term $(i-1)\Delta\cos\psi$ to the CSI phase of the ith transmit antenna. We can re-write the phase of h_i as $\angle h_i = \Phi_i - 2\pi(i-1)\Delta\cos\psi$, where Φ_i is the CSI phase caused by multi-path channel propagation. Our interest is to get the CSI phase $\angle h_i$ for each sub-carrier.

In real-world WiFi systems, the sampling clocks and carrier frequencies of the transmitter and receiver are not synchronized. This leads to Sampling Time Offset (STO) and Sampling Frequency Offset (SFO) which introduce random phase shifts. The measured CSI phase of the kth sub-carrier of the ith transmit antenna is

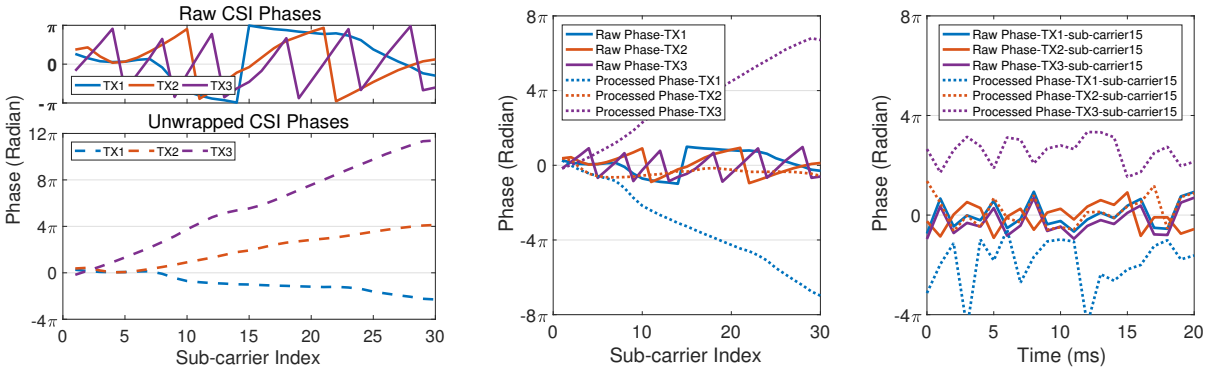
$$\Theta_{i,k} = \angle h_{i,k} - 2\pi f_\delta(k-1)\xi = \Phi_{i,k} - 2\pi(i-1)\Delta\cos\psi - 2\pi f_\delta(k-1)\xi, \tag{3}$$

where $\Phi_{i,k}$ is the CSI phase caused by multi-path channel propagation, f_δ is the frequency spacing between two consecutive sub-carriers, and ξ is the phase offset caused by STO and SFO. As shown in Fig. 6a, the unwrapped CSI phases of each transmit antenna have different slopes caused by the term $(i-1)\Delta\cos\psi$. We estimate ξ by minimizing the linear fitting error across K sub-carriers and N transmit antennas

$$\widehat{\xi} = \arg\min_{\omega} \sum_{i,k} (\Theta_{i,k} + 2\pi(i-1)\eta + 2\pi f_\delta(k-1)\omega + \beta)^2, \tag{4}$$

where η, ω and β are the fitting variables of multiple linear regression. The pre-processed CSI phase after removing random phase shifts is

$$\widehat{\angle h_{i,k}} = \Theta_{i,k} + 2\pi f_\delta(k-1)\widehat{\xi}. \tag{5}$$



(a) Raw and Unwrapped CSI Phases (b) CSI Phase vs. sub-carrier Index (c) CSI Phase vs. Sampling Time

Fig. 6. Raw CSI measurements do not capture how CSI phases change over sub-carriers and sampling time.

Since the measured CSI phases are wrapped in the range of $[-\pi, \pi]$, raw CSI measurements give wrong information about how CSI phases change over sub-carriers and sampling time. The pre-processed CSI phases are unwrapped to recover the lost information. As shown in Fig. 6b, raw CSI phases change periodically from $-\pi$ to π, while pre-processed CSI phases change nearly linearly in a wider range. Similarly, CSI phase variations over time are also corrected after CSI pre-processing. As shown in Fig. 6c, raw CSI phases of the first and second transmitting antenna change similarly, but they have very different changing patterns for pre-processed CSI phases. Raw CSI phases fail to capture the impact of the arrangement of the transmit antenna array. They give redundant information about how CSI phases change. The pre-processed CSI phases recover the information about how CSI phases change over sub-carriers and sampling time. The pre-processed CSI phases can be used by other CSI-based sensing applications.

3.2 Gesture Recognition Algorithm

SignFi uses a 9-layer CNN as the classification algorithm. CNNs are able to automatically learn parameters and features to find effective solutions for complex problems. Besides, CNNs are very fast to run in the inference stage even when the number of classes is very large. A neural network can be organized into multiple layers. The ith layer of a n-layer neural network is given by

$$\boldsymbol{y}^{(i)} = g^{(i)}\left(\boldsymbol{W}^{(i)}\boldsymbol{x}^{(i)} + \boldsymbol{b}^{(i)}\right), \tag{6}$$

where $\boldsymbol{y}^{(i)}$ is the output, $\boldsymbol{x}^{(i)}$ is the input, $\boldsymbol{W}^{(i)}$ is the weight matrix, $\boldsymbol{b}^{(i)}$ is the bias vector, and $g^{(i)}$ is the activation function [10]. The output of the previous layer is the input of the current layer, i.e., $\boldsymbol{x}^{(i)} = \boldsymbol{y}^{(i-1)}$. For the first layer, $\boldsymbol{x}^{(1)} = \boldsymbol{x}$ is the original input. For the last layer, $\boldsymbol{y}^{(n)} = \boldsymbol{y}$ is the final output. For classification problems, $\boldsymbol{y}$ contains labels in corresponding to the input $\boldsymbol{x}$. A CNN is simply a neural network with at least one of its layers involving convolution operations.

Neural networks learn the weights $\boldsymbol{W}$ and biases $\boldsymbol{b}$, using an optimization algorithm, at each layer to minimize the cost function. SignFi uses Stochastic Gradient Descent with Momentum (SGDM) to update the weights and biases. It takes small steps in the direction of the negative gradient of the loss function:

$$\theta_{l+1} = \theta_l - \alpha\nabla E(\theta_l) + \gamma(\theta_l - \theta_{l-1}), \tag{7}$$

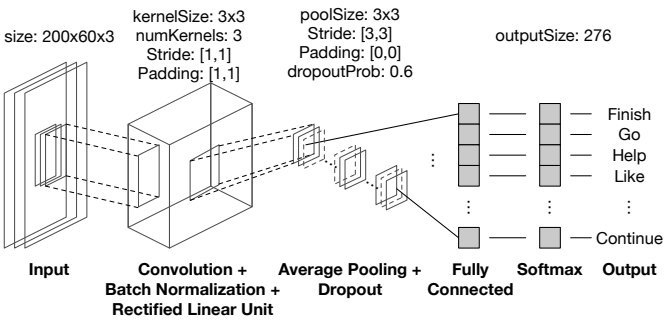


Fig. 7. Architecture and parameter settings of the 9-layer CNN of SignFi.

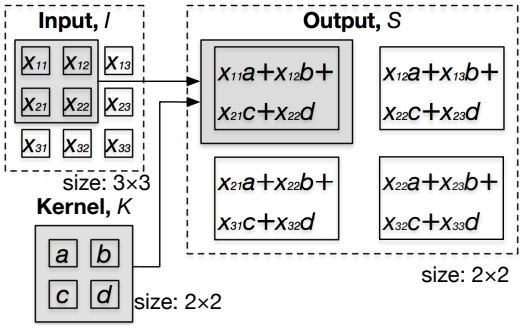


Fig. 8. An example of two-dimensional convolution with a 2×2 kernel and stride of 1, reproduced from [10].

where θ is the parameter vector, l is the iteration index, α is the learning rate, $E(\theta)$ is the loss function, and γ is the momentum term [10]. The momentum term γ controls the contribution of the previous gradient step to the current iteration. SignFi uses a momentum term of 0.9 and a learning rate of 0.01. To prevent overfitting, SignFi uses L2 regularization to add a regularization term for the weights to the loss function $E(\theta)$. The regularized loss function is

$$E_R(\theta) = E(\theta) + \lambda\Omega(\boldsymbol{W}), \tag{8}$$

where λ is the regularization factor, and $\Omega(\boldsymbol{W}) = \boldsymbol{W}^T\boldsymbol{W}/2$ is the regularization function. The regularization factor of SignFi is 0.01. Fig. 7 shows the architecture and parameter settings of the 9-layer CNN used in SignFi.

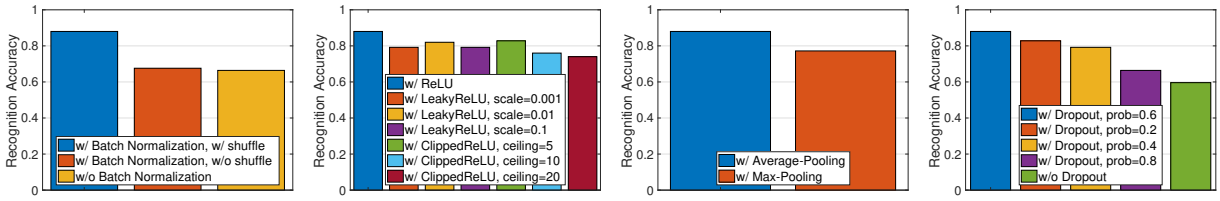
3.2.1 Input Layer. The input layer converts pre-processed CSIs of each sign gesture into a multi-dimensional tensor, which is the input format required by the CNN. This layer does not learn any parameters; it just prepares data input for the following layers. For SignFi, the size of each CSI matrix is $size(csi) = (1, 3, 30)$. There are 200 CSI samples for each sign gesture, so the size of CSI trace for each sign gesture is $(3, 30, 200)$. The CSI amplitude and phase, each with size of $(3, 30, 200)$, of each sign gesture are combined and reshaped to a tensor of size $(200, 60, 3)$ by the input layer.

3.2.2 Convolutional Layer. The convolutional layer replaces matrix multiplications with convolution operations. SignFi uses two-dimensional convolution:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n), \tag{9}$$

where I is the input, and K is the kernel [10]. Fig. 8 shows an example of two-dimensional convolution with a 2×2 kernel. The convolutional layer divides the input into multiple regions. Within each region, it computes a dot product of the input with some weights. The matrix containing the weights is called a kernel. The convolutional layer goes through the input vertically and horizontally with the same kernel. The step size of the convolutional layer moves each time is called a stride. SignFi uses three 3×3 kernels and stride of 1 in both vertical and horizontal directions. To preserve the output size of the convolutional layer and ensure all inputs are used for the same number of times, SignFi uses a padding of 1 in both vertical and horizontal directions. It pads a column/row of zeros around the edges of the original input.

The number of kernels controls the number of channels in the output of the convolutional layer. For each input region, the convolutional layer adds a bias term to the dot product of the input and the kernel. The kernel, along with its bias term, is also called a feature map. The convolutional layer learns the feature maps while



(a) Batch Normalization Layer (b) ReLU Layer (c) Pooling Layer (d) Dropout Layer

Fig. 9. Impact of batch normalization, ReLU, pooling, and dropout on recognition accuracy using leave-one-subject-out validation for 25 sign gestures and 5 users. Each user repeats 10 times for each of the 25 sign gestures.

going through the input. Since the convolution layer shares the same feature map for multiple input regions, it significantly reduces computation overhead for both training and testing. Convolutional layers are very effective and widely used in complex problems such as computer vision and natural language processing tasks. The impact of convolution on recognition accuracy of SignFi is shown later in Section 4.3.1.

3.2.3 Batch Normalization Layer. Batch normalization is used to speed up network training and reduce the sensitivity to network initialization. It makes the optimization problem easier. This allows a larger learning rate, making the network training much faster. It also improves generalization of the neural network when the training dataset contains data from different users. It first normalizes its inputs x_i over a mini-batch for each input channel. The normalized activation is

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{10}$$

where μ_B and σ_B are the mean and variance of the mini-batch [14]. In case of near-zero variances, a very small number ϵ, which is 10^{-6} in SignFi, is used to improve numerical stability. The output of the batch normalization layer is

$$y_i = \kappa \hat{x}_i + \rho, \tag{11}$$

where κ is the scale factor, ρ is the offset, and $\hat{x}_i$ is the normalized activation in equation (10) [14]. Both κ and ρ are learnable parameters that are updated during training. To take full advantage of batch normalization, SignFi shuffles the training data after each training epoch.

Fig. 9a shows the impact of batch normalization on recognition accuracy. The batch normalization layer helps prevent overfitting when the network sees a new user's data that is not shown in the training stage. Without batch normalization, the neural network tends to overfit: the recognition accuracy is only 66% while training accuracy is nearly 100%. With batch normalization but without shuffling the training data, the recognition accuracy only improves by 1%. Batch normalization along with shuffling improves recognition accuracy by 22%.

3.2.4 ReLU Layer. The Rectified Linear Unit (ReLU) layer provides fast and effective training for deep neural networks, since its activation function is easy to compute and optimize. It has been shown more effective than traditional activations, such as logistic sigmoid and hyperbolic tangent, and is widely used in CNNs [10]. The ReLU layer performs a threshold operation to each input, where any input value less than zero is set to zero, as shown in equation (12). The size of the input is not changed after the ReLU layer.

ReLU:

$$g(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \quad (12)$$

Leaky ReLU:

$$g(x) = \begin{cases} x & \text{if } x \geq 0 \\ scale * x & \text{if } x < 0 \end{cases} \quad (13)$$

Clipped ReLU:

$$g(x) = \begin{cases} ceiling & \text{if } x > ceiling \\ x & \text{if } 0 \leq x \leq ceiling \\ 0 & \text{if } x < 0 \end{cases} \quad (14)$$

There are some modified ReLUs, like leaky ReLU in equation (13) and clipped ReLU in equation (14), but they have lower recognition accuracy than ReLU in our experiments. As shown in Fig. 9b, ReLU has 6% to 9% higher accuracy than leaky ReLU and 6% to 14% higher accuracy than clipped ReLU. The possible reason is that leaky ReLU introduces some noises when $x < 0$ and clipped ReLU loses some useful information when $x > ceiling$.

3.2.5 Average-pooling Layer. The average-pooling layer reduces the number of connections to the following layers by down-sampling. It returns the average of the inputs within a rectangular region. The pooling size of SignFi is 3×3. Since there is no weight or bias, it does not provide any learning abilities. The major goal of average-pooling is to reduce the number of parameters to be learned in the following layers. It also helps reduce overfitting. Max-pooling returns the maximum, instead of the average, of selected inputs, but it has 10% lower recognition accuracy than average-pooling, as shown in Fig. 9c.

The convolutional layer, ReLU layer, and average-pooling layer are usually combined into one unit. There could be multiple of these units connecting with each other for large and complex datasets. We tried two and three of these units in our experiments, but get much lower recognition accuracy than using one unit.

3.2.6 Dropout Layer. The dropout layer is used to prevent overfitting. It randomly replaces a portion of its inputs with zero. In other words, it drops some randomly selected inputs, with a given dropout probability, and all the corresponding connections during training. As shown in Fig. 9d, the dropout layer with dropout probability of 0.6 improves recognition accuracy from 59% to 88%. Fig. 10 shows an example of the training and testing process for SignFi with and without dropout. SignFi without dropout tends to overfit, since the training accuracy reaches 100% while the testing accuracy remains around 50% and does not increase much after the 100th iteration. Similar to the average-pooling layer, the dropout layer does not provide any learning abilities.

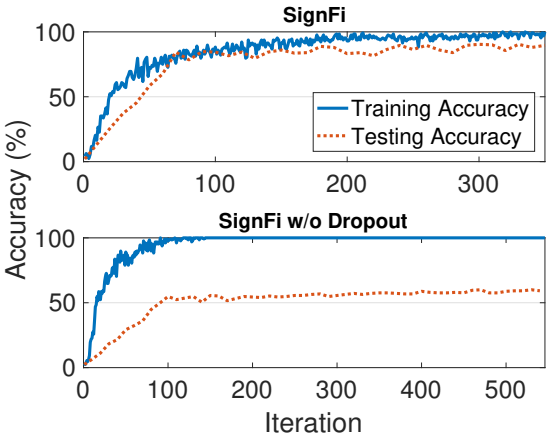
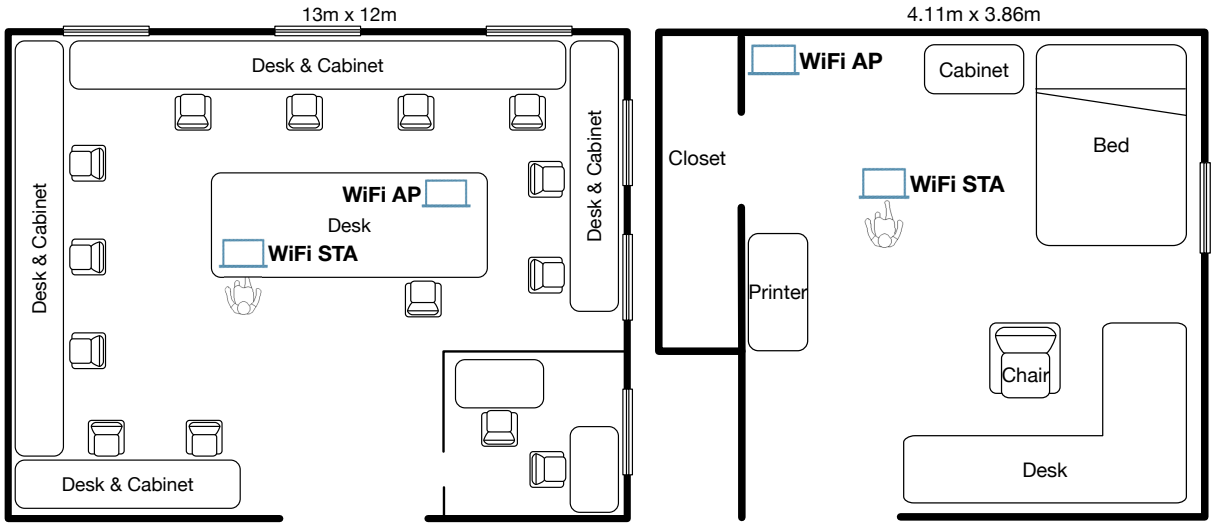


Fig. 10. Training and testing accuracy. SignFi w/o dropout tends to overfit; there is a huge gap between training accuracy and testing accuracy.

3.2.7 Fully-connected Layer. The fully-connected layer connects all of its neurons to the neurons in the previous layer, i.e., the dropout layer. The effect is to combine all the features learned by previous layers to classify the input. The size of fully-connected layer is equal to the number of all possible classes, i.e., 276 in our experiments.

3.2.8 Softmax Layer. A softmax layer and then a classification layer must follow the fully-connected layer for classification problems. The softmax layer applies the softmax function to the last fully connected layer:

$$P(c_r|x,\theta) = g(a(x,\theta))_r = \frac{e^{a_r(x,\theta)}}{\sum_{j=1}^{k} e^{a_j(x,\theta)}} = \frac{P(x,\theta|c_r)P(c_r)}{\sum_{j=1}^{k} P(x,\theta|c_j)P(c_j)}, \quad (15)$$



(a) Lab Environment (b) Home Environment

Fig. 11. Floor plan and measurement settings of the lab and home environments.

where $g(a)_r = e^{a_r}/\sum_{j=1}^{k} e^{a_j}$ is the softmax function with $0 \leq g(a)_r \leq 1$ and $\sum_{j=1}^{k} g(a)_j = 1$. Moreover, $a_r(x, \theta) = \ln(P(x, \theta|c_r)P(c_r))$, where $P(x, \theta|c_r)$ is the conditional probability of the given class r, $P(c_r)$ is the class prior probability, and θ is the parameter vector.

3.2.9 Classification Layer. The classification output layer takes the values from the softmax function and assigns each input to one of the k mutually exclusive classes using the cross entropy function

$$E(\theta) = -\sum_{i=1}^{n}\sum_{j=1}^{k} t_{ij} \ln y_j(x_i, \theta), \tag{16}$$

where t_{ij} represents that the ith sample belongs to the jth class, and θ is the parameter vector. $y_j(x_i, \theta)$ is the output for the ith sample, which is the value from the softmax function. It represents the probability that the network associates the ith input with class j, i.e., $P(t_j = 1|x_i)$.

4 EVALUATION

In this section, we first give the experiment setup, including measurement layout and displacements, data collection procedure, and WiFi settings. We compare SignFi with existing classification algorithms in different environments. Two performance metrics, recognition accuracy and time consumption of training and testing, are evaluated. We also check the impact of convolution, signal processing, and sampling rate on the recognition accuracy of SignFi. Finally, we run user independence test for 150 sign gestures performed by 5 different users.

4.1 Experiment Setup

We collect CSI traces for 276 sign gestures that are frequently used in daily life. CSI traces are measured in both lab and home environments. Fig. 11 shows the floor plan and measurement settings for the lab and home environments. The dimension of the lab and home is 13m×12m and 4.11m×3.86m, respectively. The lab has more

Table 1. Number of Sign Words in Different Categories used in the Experiments

Common	Animals	Colors	Descriptions	Family	Food	Home	People	Questions	School	Time	Others	Total
16	15	12	32	31	54	17	13	6	26	31	23	276

Table 2. Data Collection Summary

	User	Age	Weight/Height	Data Collection Date (Number of Signs × Number of Repetitions)	Gesture Duration	Number of Instances
Lab	User 1	39	90kg/170cm	Oct. 18, 2017 (25×10); Nov. 2, 2017 (125×10)	1s-2.5s	1,500
	User 2	28	61kg/174cm	Oct. 18, 2017 (25×10); Oct. 30, 2017 (125×10)	0.5s-1.5s	1,500
	User 3	31	55kg/168cm	Oct. 21, 2017 (25×10); Nov. 6, 2017 (125×10)	0.5s-1.5s	1,500
	User 4	26	65kg/180cm	Oct. 23, 2017 (25×10); Oct. 31, 2017 (125×10)	1s-2.5s	1,500
	User 5[a]	29	68kg/171cm	Jul. 18, 2017 (166×20); Jul. 19, 2017 (110×20)	0.5s-1.5s	5,520
Home	User 5	29	68kg/171cm	Jun. 8, 2017 (32×10); Jun. 25, 2017 (68×10); Jul. 4, 2017 (100×10); Jul. 11, 2017 (25×10); Jul. 12, 2017 (51×10)	0.5s-1.5s	2,760

[a] Compared with user 1 to 4, user 5 has different experiment settings, such as laptop displacement, surrounding objects, desk and chair arrangements, etc., even though they are in the same lab environment. The data collection time of user 5 is 3-4 months earlier than that of user 1 to 4, so it is hard to recover the same settings.

surrounding objects, leading to a more complex multi-path environment than the home. The distance between the AP and STA is 230cm and 130cm, respectively, for the lab and home environment. For the home environment, the transmit antenna array is orthogonal to the direction from the AP to STA. For the lab environment, the angle between the transmit antenna array and the direct path is about 40 degrees. The major differences of these two environments are: (1) dimension of the room, (2) distance between the AP and STA, (3) angle between the transmit antenna array and the direct path, and (4) multi-path environments.

Table 1 summaries the 276 sign words used in our experiments divided into different categories. We select the 253 basic sign words from [35]. These sign words are the most important words for ASL beginners and are frequently used in daily life. Some sign words have different gestures; we only select one of the gestures that have the same meaning. We do not select compound signs that are composed of more than three signs. For comparison, we also run experiments on 25 sign gestures from [21]. Two sign words, "phone" and "kitchen", are already included in the 253 basic sign gestures. In total, 276 sign gestures are tested in our experiments.

We collect CSI traces from 5 male users who do not know how to sign before the experiments. Table 2 shows the summary of data collection. Different users may have different gesture durations and slightly different hand/finger movements for the same sign word. For user 1 to 4, we collect CSI traces only in the lab environment. Each of the 4 users makes 150 sign gestures with each gesture repeated for 10 times. There are 6,000 gesture instances for these 4 users. For user 5, we collect CSI traces in both the lab and home environments. Each sign gesture has 20 instances for the lab environment and 10 instances for the home environment. In total, there are 8,280 gesture instances for user 5. CSI traces, labels, and videos of the 276 sign words are available for download[2].

Fig. 12 shows the experiment setup in the lab environment. During the experiments, each user first watches video on [35] to learn how to sign for one word. As long as the user feels comfortable to conduct the sign gesture smoothly, we begin to collect CSI traces for this sign gesture. The user repeats the sign gesture in front of a WiFi Station (STA), which exchanges packets with a nearby WiFi Access Point (AP). The user begins to make the

[2]https://yongsen.github.io/SignFi/

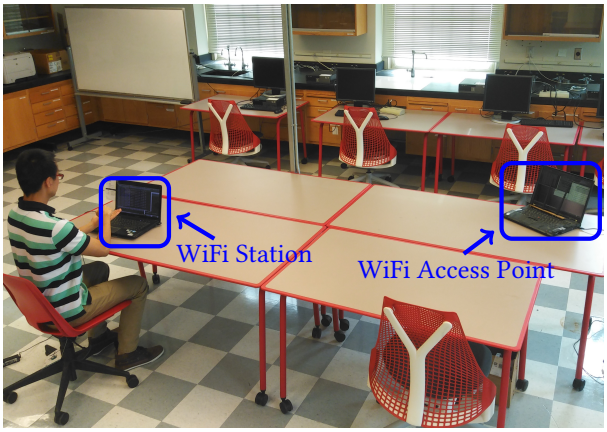


Fig. 12. Experiment setup for the lab environment.

Table 3. Classification Algorithm and CSI Size for each Label in Fig. 13, 14, and 15

Label	Algorithm	CSI Size
kNN+DTW+5subc	kNN+DTW	(1,1,5)
kNN+DTW+5subc+MIMO	kNN+DTW	(1,3,5)
kNN+DTW+30subc	kNN+DTW	(1,1,30)
kNN+DTW+30subc+MIMO	kNN+DTW	(1,3,30)
CNN+5subc	CNN	(1,1,5)
CNN+30subc	CNN	(1,1,30)
SignFi	CNN	(1,3,30)

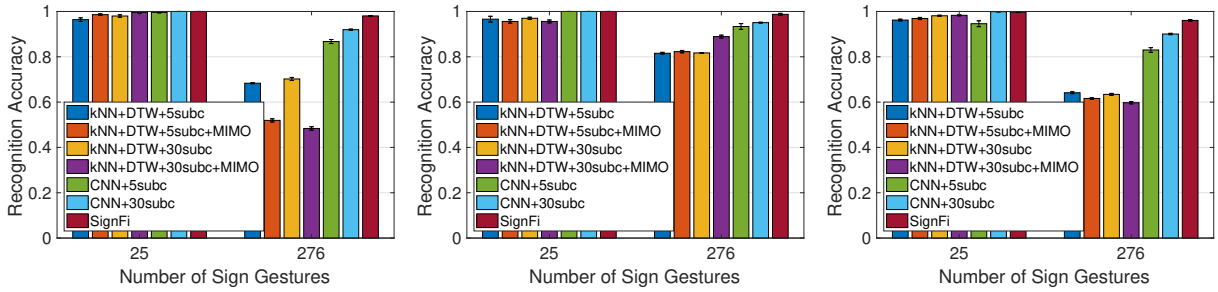
first sign as seeing "Sign Starts ... 1" on the screen of the STA. At the same time, the AP sends 802.11n packets periodically to the STA. The STA collects CSI measurements while the person is making sign gestures. The user repeats the same sign gesture until the screen of the STA shows "Sign Starts ... [n]". Here n could be 11 or 21 depending on whether 10 or 20 gesture instances are collected. We repeat this procedure, i.e., watching the video, repeating the sign gesture, and collecting CSI traces, for all the sign gestures.

The WiFi AP and STA are two laptops with Intel WiFi Link 5300 installed. CSI measurements are collected using openrf [16], which is modified based on the 802.11n CSI tool [12]. The WiFi AP and STA operate at 5GHz, and the channel width is 20MHz. Note that the 802.11n CSI tool only provides CSI values of 30 sub-carriers even though a 20MHz WiFi channel has 52 sub-carriers. The AP has 3 external antennas, and the STA has 1 internal antenna. The transmitting power is fixed at 15dBm. All experiments are conducted in the presence of other WiFi signals. As shown in Fig. 12, the user is not on the direct path between the STA and AP. This is a normal case in real life. It also makes sign gesture recognition much harder, since the strength of signals reflected by the person is much lower than that of the direct path signals. Training and testing are performed by a Linux desktop with an 8-core i7-4790 CPU at 3.60GHz and 15.6GB of RAM.

4.2 Comparing SignFi with Existing Methods

We compare SignFi with the classification algorithm kNN with DTW used in [18, 21]. The input signals used in [21] are the Received Signal Strength (RSS) and the amplitude and phase of 5 sub-carriers with the least average cross-correlation. We feed the same input signals to kNN with DTW. CSI traces of different transmit antennas provides different results for kNN with DTW. We select the transmit antenna that has the highest recognition accuracy. We also use CSI traces from all the transmit antennas and all the 30 sub-carriers as input signals. To check the impact of input signals on CNN, we also run CNN using CSI traces from 5 or 30 sub-carriers. Table 3 gives a summary of the classification algorithm and CSI size for each label used in our comparison. In this section, we only use the data of user 5. We run 5-fold cross validation using 5,520 instances from the lab, 2,760 instances from the home, and 8,280 instances from the lap+home environment.

4.2.1 Recognition Accuracy. Recognition accuracy is defined as the number of correctly classified instances divided by the number of all testing instances. Fig. 13 shows recognition accuracy results in different environments. SignFi provides high recognition accuracy for all the datasets. For the 276 sign gestures, the average recognition



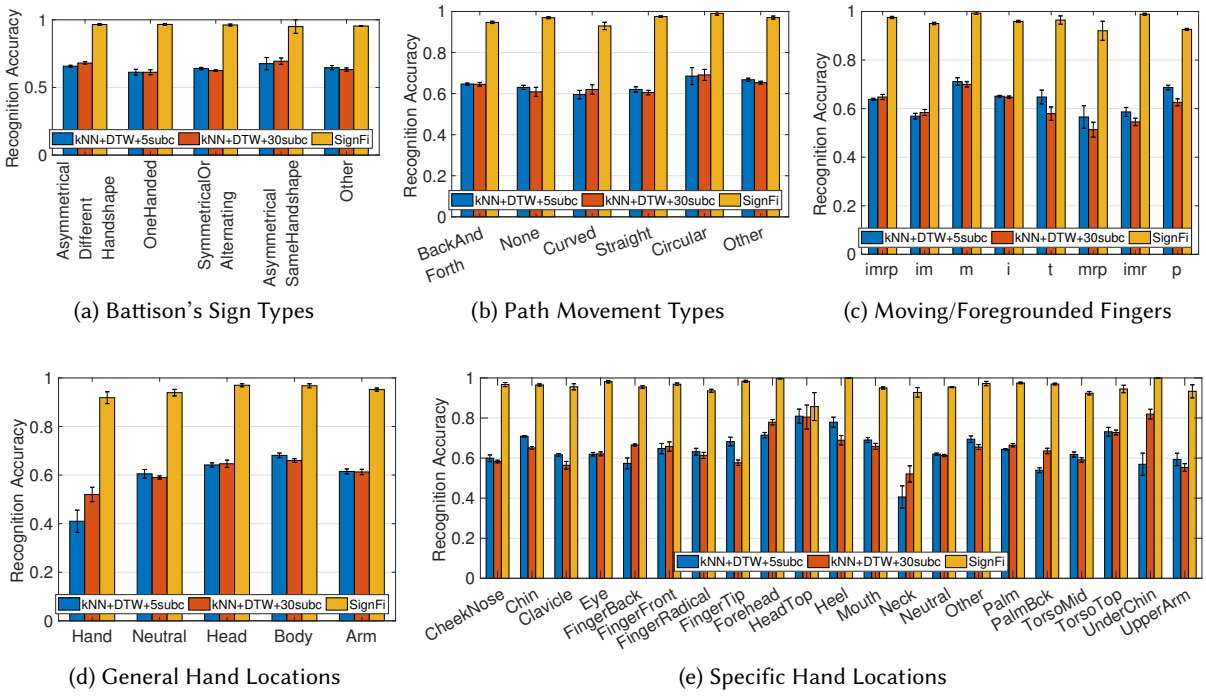
(a) Lab Environment (b) Home Environment (c) Lab+Home Environment

Fig. 13. Recognition accuracy in different environments.

accuracy of SignFi is 98.01%, 98.91%, and 94.81% for the lab, home, and lap+home environment, respectively. With only 5 sub-carriers of non-MIMO CSI traces, CNN has average accuracy of 80.74%, 93.37%, and 83.00% for the lab, home, and lab+home environment, respectively. Using the same input, with either 5 or 30 sub-carriers of non-MIMO CSI traces, CNN has much higher accuracy than kNN with DTW in the lab and lab+home environments. For the lab+home environment, SignFi still has 94.81% accuracy even though the lab and home have very different experiment settings.

For the 25 sign gestures, all classification algorithms with different input signals have over 95% recognition accuracy for all the three environment settings. For the 276 sign gestures, the accuracy of kNN with DTW decreases dramatically for the lab and lap+home environments. For the lab environment, the average recognition accuracy is only 68.33% and 70.20% for kNN with DTW using CSI traces of 5 and 30 sub-carriers, respectively. Adding MIMO CSI measurements further decreases the average accuracy to 51.93% and 48.30%. The major reason is that the lab has a complex multi-path environment, which heavily impacts MIMO. Another reason is that the distance between the WiFi AP and STA in the lab is longer than that of the home environment, so the strength of reflected signals is lower. This leads to more noise signals for the lab environment. The average accuracy of kNN with DTW is increased to 82% for the home environment. Using both MIMO and 30 sub-carriers of CSI traces, the accuracy of kNN with DTW is improved to 88.90%. The reason is that the home environment has less multi-path signals and shorter distance between the AP and STA. Even though kNN with DTW provides 88.90% accuracy using MIMO and 30 sub-carriers of CSI traces, it takes extremely long time to finish the testing. The time consumption of training and testing is shown later in the next subsection.

To get a better understanding about sign gesture recognition, we break down the 276 sign gestures into different categories and check the recognition accuracy in each category, as shown in Fig. 14. All the evaluation results in Fig. 14 are from the lab+home environment. The number of sign gestures in each group of each category for the 276 sign gestures is shown in Fig. 3 in Section 2. All the sign gestures in the same group of each category has similar patterns. The evaluation results show that: (1) whether the classification algorithms can distinguish similar sign gestures; (2) what kinds of sign gestures are hard to recognize. SignFi has high accuracy for all groups in each category. This means that even for sign gestures with very similar patterns, SignFi is still able to distinguish them from each other. For sign types in Fig. 14a and path movement types in Fig. 14b, there is no significant difference for each group. For Fig. 14c, kNN with DTW has the lowest accuracy when there are three moving/foregrounded fingers, "mrp" and "imr". For general hand locations in Fig. 14d, sign gestures with the dominate hand near the non-dominate hand are the hardest to recognize, for both SignFi and kNN with DTW. For general hand location of "Hand" and specific hand location of "Neck", the recognition accuracy of kNN with



(a) Battison's Sign Types (b) Path Movement Types (c) Moving/Foregrounded Fingers

(d) General Hand Locations (e) Specific Hand Locations

Fig. 14. Recognition accuracy in different categories for the lab+home environment (8,280 gesture instances for 276 signs).

DTW is only 40% and 51% using 5 and 30 sub-carriers' CSI, respectively. For specific hand location of "HeadTop", kNN with DTW has comparable accuracy as SignFi, as shown in Fig. 14e. For all other groups in each category, kNN with DTW has much lower accuracy than SignFi.

4.2.2 Time Consumption of Training and Testing. Fig. 15 shows the time consumption of training and testing of different classification algorithms with different CSI inputs. The testing time of SignFi is much shorter than that of kNN with DTW. For kNN with DTW, the testing time of each sign gesture is 33.36ms using 5 sub-carriers of non-MIMO CSI traces. It proliferates to 5,147.20ms if the input has 30 sub-carriers of MIMO CSI traces. The testing time of SignFi is 0.62ms. However, SignFi does have longer training time than kNN with DTW. The maximum training time for kNN with DTW is only 0.046ms. SignFi takes 8.28ms for CSI processing and CNN training for each sign gesture. It takes 0.05ms and 2.12ms for CNN to finish training using 5 and 30 sub-carriers of non-MIMO CSI traces, respectively. Since training usually can be performed offline and testing must be done in real-time, it is more important to reduce the testing time. Therefore, SignFi is more practical than kNN with DTW to be implemented in real-time.

4.3 More Discussions on SignFi

In this section, we investigate the impact of convolution, signal processing, and sampling rate on the recognition accuracy of SignFi. We run 5-fold cross validation using the data of 276 sign gestures from user 5.

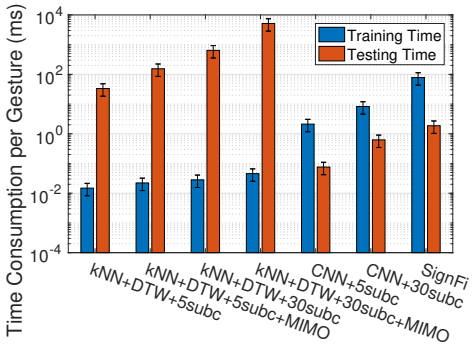
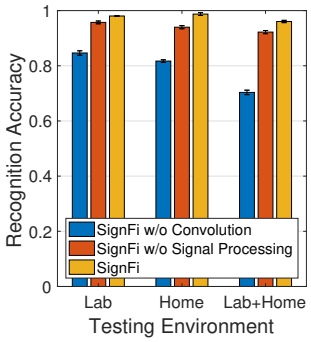
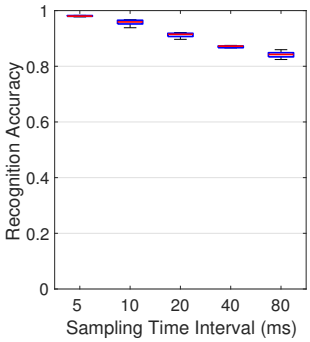


Fig. 15. Time consumption of training and testing for each sign gesture.



(a) Impact of convolution and signal processing



(b) Impact of sampling rate

Fig. 16. Impact of convolution, signal processing, and sampling time interval of SignFi.

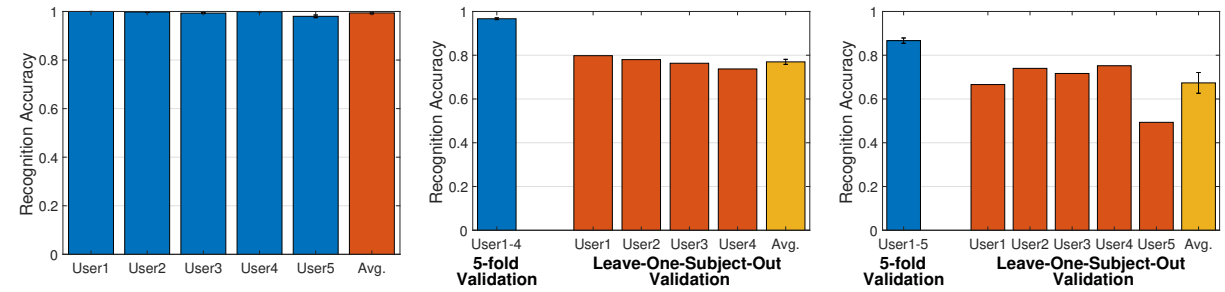
4.3.1 Impact of Convolution. The reason we use CNN as the classification algorithm is that CNN has higher recognition accuracy than other neural networks that do not have the convolutional layer. As shown in Fig. 16a, the recognition accuracy of SignFi without convolution is 84.64%, 81.70%, and 70.34% for the lab, home, and lab+home environment, respectively. For the lab+home environment, the accuracy improvement due to convolution is 24.47%, which is the highest among all the three environments. The evaluation results show that the convolution layer has a significant impact on the recognition accuracy of SignFi.

4.3.2 Impact of Signal Processing. We also check the impact of SignFi signal processing that removes noises from raw CSI measurements, as shown in Fig. 16. Without SignFi signal processing, the average recognition accuracy is 95.72%, 93.98%, and 92.21% for the lab, home, and lab+home environment, respectively. SignFi signal processing has the highest accuracy improvement, which is 4.93%, for the lab environment. The reason is that SignFi signal processing only removes random phase offsets. It does not filter out other noise signals. The layout, surrounding environment, and displacement of the AP and STA of the lab and home are very different, leading to very different noise signals.

4.3.3 Impact of Sampling Rate. Another import factor that influences the recognition accuracy of SignFi is the CSI sampling rate. For all the previous evaluation results, the WiFi STA measures CSI about every 5ms. We change the sampling time interval for the CSI dataset collected from the lab environment, and run SignFi with 5-fold cross validation again. The evaluation results are shown in Fig. 16b. When the sampling time interval increases from 5ms to 10ms, the average recognition accuracy decreases from 98.01% to 95.72%. SignFi still has high recognition accuracy using 10ms of sampling interval, considering there are 5,520 instances of 276 sign gestures. When the sampling time interval increases to 20ms, the average recognition accuracy decreases to 91.12%. Based on these results, the sampling time interval should be no larger than 20ms to get high recognition accuracy. For sampling interval of 40ms and 80ms, the average accuracy further decreases to 87.05% and 84.17%.

4.4 User Independence Test

This section gives user independence test using CSI traces of 150 sign gestures from 5 different users. There are 7,500 gesture instances in total. We run self test, 5-fold cross validation, and leave-one-subject-out validation



(a) Self Test, 5-fold Validation for Each User (b) Cross Validation for User 1 to 4 (c) Cross Validation for User 1 to 5

Fig. 17. Recognition accuracy of user independence test.

using CSI traces from all the 5 users. For self test, wherein training and testing only include CSI traces from the same user, the recognition accuracy for each user is above 98%, as shown in Fig. 17a. In Section 4.4.1, we use CSI traces of just user 1 to 4 to separate the impact of user 5. As shown in Table 2, CSI traces of user 5 were collected three to four months before user 1 to 4. We were unable to use the exact same experiment settings, such as laptop displacement, surrounding objects, desk and chair arrangements, etc., for user 1 to 4 and user 5. User 1 to 4 have almost the same experiment settings, which are different from that of user 5. We also show the impact of different data collection dates and settings by including CSI traces of user 5 in Section 4.4.2.

4.4.1 Users with Similar Data Collection Dates and Settings. We first run 5-fold cross validation using CSI traces of user 1 to 4. In other words, CSI traces from different users are mixed together and randomly divided into training and testing datasets. The average recognition accuracy is 96.68%, as shown in Fig. 17b. This means that SignFi is robust to different users, considering these users have different body sizes and gesture durations. For leave-one-subject-out cross validation, the recognition accuracy is in the range of 73.73% and 79.80%, and the average recognition accuracy is 76.96%. Comparing with 5-fold cross validation, the recognition accuracy of leave-one-subject-out cross validation decreases by 20%.

4.4.2 Users with Different Data Collection Dates and Settings. When CSI traces of user 5 are included, the recognition accuracy of 5-fold cross validation decreases to 86.66%, as shown in Fig. 17c. This means that experiment settings have a significant impact on SignFi. For leave-one-subject-out validation, the average accuracy drops to 67.36%. The accuracy of user 1 to 3 decreases by 13.2%, 4%, and 4.7%, respectively. The recognition accuracy of user 5 is only 49.3%. This is not adequate for practical usage, but is still very good compared to a random algorithm, which only has 1/150=0.06% accuracy. From the evaluation results, we can see that different users and different experiment settings have a great impact on SignFi. When there is a new user, we may need the user first conduct some sign gestures to train the neural network to get a good recognition accuracy.

5 RELATED WORK

There are many sign language recognition systems using different signals. We give a comparison of different sign language recognition technologies in Table 4. Since sign language recognition needs gesture recognition, we give a summary of gesture recognition technologies in Table 5.

Table 4. Comparison of Sign Language Recognition Technologies

Comparison Technologies	Signal/Device Used	Intrusive?	Granularity	Gesture Type	Recognition Algorithm[a]	Number of Sign Gestures	Recognition Accuracy
Zafrulla2011 [47]	Kinect, gloves and sensors	Yes	Hand/Finger	Static	HMM	26	51.5% (seated); 76.12% (standing)
Sun2015 [36]	Kinect	No	Hand/Finger	Dynamic	SVM	73	86.0%
Pigou2015 [28]					CNN	20	91.7%
Huang2015 [13]					CNN	25	94.2%
Chuan2014 [6]	Leap Motion	No	Finger	Static	kNN; SVM	26	72.78% (kNN); 79.83% (SVM)
Quesada2015 [30]					SVM	10	79.17%
Funasaka2015 [8]					SVM	26	82.71%
Mapari2016 [20]					MLP	26	90%
Naglot2016 [24]					MLP+BP	26	96.15%
DeepASL [7]	Leap Motion	No	Hand/Finger	Dynamic	RNN	56	94.5%
Savur2015 [32]	sEMG Sensor	Yes	Finger	Static	SVM	26	91% (offline); 82.3% (real-time)
WiSign [33]	WiFi (2.4/5 GHz)	No	Hand	Dynamic	SVM	5	93.8%
WiFinger* [18]			Finger	Static	kNN+DTW	9	90.4%
Melgarejo2014 [21]			Hand/Finger	Dynamic	kNN+DTW	25 (wheelchair); 14 (car)	92% (wheelchair); 84% (car)
SignFi (Our design)	WiFi (5 GHz)	No	Head/Arm/ Hand/Finger	Dynamic	CNN	**276**	**98% (lab); 98% (home); 94% (lab+home)**

[a] HMM: Hidden Markov Model; SVM: Support Vector Machine; CNN: Convolutional Neural Network; kNN: k Nearest Neighbor; MLP: Multi-Layer Perceptron; BP: Back-Propagation; RNN: Recurrent Neural Network; DTW: Dynamic Time Warping

5.1 Sign Language Recognition

A brief summary of sign language recognition technologies is given in Table 4. There are many vision-based sign language recognition systems using cameras [34] or the Kinect sensor [13, 28, 36, 47]. For example, the SignAll prototype [34] uses 3 cameras and 1 depth sensor to track hand gestures. A Kinect sensor, along with color gloves and accelerometers, are used to recognize 26 alphabet letters in [47]. Only the Kinect sensor is used in [13, 28, 36] to recognize sign gestures. Paper [36] is able to recognize 73 basic signs with 86.0% accuracy. These vision-based systems are subject to lighting conditions. Recently, many papers use the Leap Motion sensor for sign language recognition [6, 8, 20, 23, 24, 30, 36, 47]. As shown in Table 4, these systems are tested on signs for numbers and alphabet letters. These signs only involve simple finger postures. Leap Motion-based systems can only recognize finger gestures, and the hands must be in a very small region near the sensor. Some sign recognition systems use sensors, like motion sensors in SignAloud [26] and surface Electromyography (sEMG) sensors in [32], but they are intrusive and require sensors to be attached on fingers.

WiFi signals are used to recognize sign gestures in a non-intrusive way in [18, 21, 33]. But they can only recognize simple ASL gestures: 5 hand gestures in [33], 9 digits finger postures in [18], and 25 hand/finger gestures in [21]. SignFi also uses WiFi signals, and it is able to recognize 276 very complex sign gestures with 97.03% accuracy. For all the existing systems in Table 4, only [36] is evaluated on a relatively large number of complex sign gestures. It is able to recognize 73 sign gestures with 86.0% accuracy. These 73 sign gestures do not include signs that look similar in vision. SignFi is able to distinguish more complex sign gestures that have very similar hand/arm/finger movements, and with higher accuracy than the Kinect-based solution in [36].

Table 5. Comparison of Gesture Recognition Technologies

Comparison Technologies	Signal/Device Used	Intrusive?	Granularity	Number of Gestures[a]	Recognition Accuracy
E-Gesture [27]	Motion Sensor	Yes	Hand	8	94.6%
Watanabe2016 [43]			Hand	15	79%
Serendipity [44]			Finger	5	87%
Xu2015 [45]			Arm/Hand/Finger	37	98%
FingerPad [3]	Magnetic Sensor	Yes	Finger	N/A	N/A (finger tracking)
uTrack [4]					
Finexus [5]					
Savur2015 [32]	sEMG Sensor	Yes	Finger	26	91% (offline); 82.3% (real-time)
SoundWave [11]	Audio (18-22 KHz)	No	Hand	5	94.7% (home); 94.3% (cafe)
AudioGest [31]			Hand	6	94.15%
FingerIO [25]			Finger	N/A	N/A (finger tracking)
Strata [46]			Finger	N/A	N/A (finger tracking)
LLAP [42]			Hand/Finger	N/A	N/A (hand/finger tracking)
SlideSwipe [49]	GSM (850 MHz)	No	Hand	14	87.2%
AllSee [15]	TV (725 MHz); RFID (915 MHz)	No	Hand/Finger	8	94.4% (TV); 97% (RFID)
RF-IDraw[41]	RFID (922 MHz)	Yes	Finger	N/A	N/A (finger tracking)
WiSee [29]	WiFi (2.4/5 GHz)	No	Body/Hand/Leg	9	94%
WiDraw [37]			Hand	N/A	N/A (hand tracking)
WiGest [1]			Hand	7	87.5% (1 AP); 96% (3 APs)
WiAG [40]			Hand	6	91.4%
WiSign [33]			Hand	5	93.8%
WiFinger** [38]			Finger	8	93%
Mudra [48]			Finger	9	96%
WiFinger* [18]			Finger	9	90.4%
Melgarejo2014 [21]			Hand/Finger	25 (wheelchair); 14 (car)	92% (wheelchair); 84% (car)
Molchanov2015 [22]	FMCW (24 GHz)	No	Finger	10	94.1%
Soli [19]	Millimeter Wave (60 GHz)	No	Finger	4	92.1%
SignFi (Our design)	WiFi (5 GHz)	No	Head/Arm/ Hand/Finger	**276**	**98% (lab); 98% (home); 94% (lab+home)**

[a] Only WiSign [33], WiFinger* [18], Melgarejo2014 [21], and SignFi (our design) evaluate on sign language gestures.

5.2 Gesture Recognition

One import part of sign language recognition is gesture recognition. Table 5 gives a comparison of gesture recognition technologies using different signals. Motion sensors, like accelerometers used in [27, 43], are widely used for hand gesture recognition. Some papers use accelerometers and gyroscopes to recognize finger gestures [44, 45]. A smartwatch with accelerometers and gyroscopes is able to measure tendons movements and identify 37 (13 finger, 14 hand and 10 arm) gestures with 98% accuracy [45]. However, finger gestures must have the wrist and arm affixed to the chair arm while hand gestures must have the arm affixed. So finger gestures involve only finger movements, and hand gestures involve only wrist movements. This is not realistic for sign gesture recognition

wherein a sign gesture may contain all hand, arm and finger movements. Magnetic sensors are used in [3–5] and sEMG sensors are used in [32] to recognize finger gestures, but these methods require sensors attached to the fingers of the signer. Sensor-based gesture recognition systems are intrusive.

Many gesture recognition systems use audio or wireless signals as the input. SoundWave [11] and AudioGest [31] use audio signals to recognize simple hand gestures with up to 95% accuracy. Audio signals are used in [25, 42, 46] to track 2-D finger movements with tracking accuracy of 8mm, 1cm, and 4.6mm, respectively. For audio-based gesture recognition, the distance between the device and the hand/finger must be very short (usually less than 20cm). There are many gesture recognition systems using wireless signals, including Global System for Mobile communications (GSM) [49], TV [15], Radio-Frequency Identification (RFID) [15, 41], WiFi [1, 18, 21, 29, 33, 37, 38, 40, 48], Frequency Modulated Continuous Wave (FMCW) [22], and millimeter wave [19]. Sign language recognition needs to distinguish finger-level gestures/postures, which are not tested by SlideSwipe [49], WiSee [29], WiDraw [37], WiGest [1], or WiAG [40]. Although other wireless-based gesture recognition systems can detect finger-level gestures, none of them are tested on more than 25 gestures. SignFi is able to recognize 276 sign gestures with above 94% accuracy using WiFi signals.

6 CONCLUSION

In this paper, we propose a sign language recognition system, SignFi, to recognize frequently used sign gestures using WiFi signals. SignFi measures Channel State Information (CSI) by WiFi packets and uses a 9-layer Convolutional Neural Network (CNN) as the classification algorithm. The average recognition accuracy of SignFi is 98.01%, 98.91%, and 94.81% for the lab, home, and lab+home environment, respectively. For 7,500 instances of 150 sign gestures performed by 5 different users, the recognition accuracy of SignFi is 86.66%.

We have shown that SignFi is robust for two environments and five users. There are many other factors that may influence the recognition performance. For example, the distance between the person and the AP/STA could be different. The direction and orientation of the person with respect to the AP/STA could also change. There could be multiple persons or other moving objects around. The person or other objects could block the direct path from the AP to STA. We plan to collect more CSI traces and run tests considering these factors. Another import task is sentence-level sign language recognition. In this paper, CSI measurements are manually segmented for each sign gesture. It introduces many challenges for sentence-level sign language recognition when CSI traces are not manually segmented. We leave automatic time segmentation to future work. Recurrent Neural Networks (RNN) with Long-Short-Term-Memory (LSTM) could help for automatic sentence-level sign language recognition.

ACKNOWLEDGMENTS

This work was supported in part by U.S. National Science Foundation under grants CNS-1253506. The authors would like to thank the anonymous reviewers and primary associate editor for their valuable comments.

REFERENCES

[1] H. Abdelnasser, M. Youssef, and K. A. Harras. 2015. WiGest: A ubiquitous WiFi-based gesture recognition system. In *2015 IEEE Conference on Computer Communications (INFOCOM)*. 1472–1480. https://doi.org/10.1109/INFOCOM.2015.7218525

[2] Naomi K. Caselli, Zed Sevcikova Sehyr, Ariel M. Cohen-Goldberg, and Karen Emmorey. 2017. ASL-LEX: A lexical database of American Sign Language. *Behavior Research Methods* 49, 2 (April 2017), 784–801. https://doi.org/10.3758/s13428-016-0742-0

[3] 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 (UIST '13)*. 255–260. https://doi.org/10.1145/2501988.2502016

[4] Ke-Yu Chen, Kent Lyons, Sean White, and Shwetak Patel. 2013. uTrack: 3D Input Using Two Magnetic Sensors. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. 237–244. https://doi.org/10.1145/2501988.2502035

[5] 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 (CHI '16)*. 1504–1514. https://doi.org/10.1145/2858036.

2858125
[6] Ching-Hua Chuan, Eric Regina, and Caroline Guardino. 2014. American Sign Language Recognition Using Leap Motion Sensor. In *Proceedings of the 2014 13th International Conference on Machine Learning and Applications (ICMLA '14)*. 541–544. https://doi.org/10.1109/ICMLA.2014.110
[7] Biyi Fang, Jillian Co, and Mi Zhang. 2017. DeepASL: Enabling Ubiquitous and Non-Intrusive Word and Sentence-Level Sign Language Translation. In *Proceedings of the 15th ACM Conference on Embedded Networked Sensor Systems (SenSys '17)*.
[8] Makiko Funasaka, Yu Ishikawa, Masami Takata, and Kazuki Joe. 2015. Sign Language Recognition using Leap Motion Controller. In *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA)*.
[9] David Goldberg, Dennis Looney, and Natalia Lusin. 2015. Enrollments in Languages Other Than English in United States Institutions of Higher Education, Fall 2013. In *Modern Language Association*.
[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.
[11] Sidhant Gupta, Daniel Morris, Shwetak Patel, and Desney Tan. 2012. SoundWave: Using the Doppler Effect to Sense Gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. 1911–1914. https://doi.org/10.1145/2207676.2208331
[12] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. 2011. Tool Release: Gathering 802.11n Traces with Channel State Information. *SIGCOMM Comput. Commun. Rev.* 41, 1 (Jan. 2011), 53–53. https://doi.org/10.1145/1925861.1925870
[13] Jie Huang, Wengang Zhou, Houqiang Li, and Weiping Li. 2015. Sign Language Recognition using 3D Convolutional Neural Networks. In *2015 IEEE International Conference on Multimedia and Expo (ICME)*. 1–6. https://doi.org/10.1109/ICME.2015.7177428
[14] Sergey Ioffe and Christian Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37 (ICML'15)*. JMLR.org, 448–456. http://dl.acm.org/citation.cfm?id=3045118.3045167
[15] Bryce Kellogg, Vamsi Talla, and Shyamnath Gollakota. 2014. Bringing Gesture Recognition to All Devices. In *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation (NSDI'14)*. 303–316. http://dl.acm.org/citation.cfm?id=2616448.2616477
[16] Swarun Kumar, Diego Cifuentes, Shyamnath Gollakota, and Dina Katabi. 2013. Bringing Cross-layer MIMO to Today's Wireless LANs. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM (SIGCOMM '13)*. 387–398. https://doi.org/10.1145/2486001.2486034
[17] H.L. Lane, R. Hoffmeister, and B.J. Bahan. 1996. *A Journey Into the Deaf-world*. DawnSignPress.
[18] Hong Li, Wei Yang, Jianxin Wang, Yang Xu, and Liusheng Huang. 2016. WiFinger: Talk to Your Smart Devices with Finger-grained Gesture. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '16)*. 250–261. https://doi.org/10.1145/2971648.2971738
[19] 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 Trans. Graph.* 35, 4, Article 142 (July 2016), 19 pages. https://doi.org/10.1145/2897824.2925953
[20] Rajesh B. Mapari and Govind Kharat. 2016. American Static Signs Recognition Using Leap Motion Sensor. In *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies (ICTCS '16)*. Article 67, 5 pages. https://doi.org/10.1145/2905055.2905125
[21] Pedro Melgarejo, Xinyu Zhang, Parameswaran Ramanathan, and David Chu. 2014. Leveraging Directional Antenna Capabilities for Fine-grained Gesture Recognition. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14)*. 541–551. https://doi.org/10.1145/2632048.2632095
[22] P. Molchanov, S. Gupta, K. Kim, and K. Pulli. 2015. Multi-sensor system for driver's hand-gesture recognition. In *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, Vol. 1. 1–8. https://doi.org/10.1109/FG.2015.7163132
[23] MotionSavvy. 2017. MotionSavvy UNI. (2017). Retrieved July 23, 2017 from http://www.motionsavvy.com/uni.html.
[24] D. Naglot and M. Kulkarni. 2016. Real time sign language recognition using the leap motion controller. In *2016 International Conference on Inventive Computation Technologies (ICICT)*, Vol. 3. 1–5. https://doi.org/10.1109/INVENTIVE.2016.7830097
[25] Rajalakshmi Nandakumar, Vikram Iyer, Desney Tan, and Shyamnath Gollakota. 2016. FingerIO: Using Active Sonar for Fine-Grained Finger Tracking. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. 1515–1525. https://doi.org/10.1145/2858036.2858580
[26] University of Washington. 2016. SignAloud Demo. (2016). Retrieved July 23, 2017 from https://youtu.be/4uY-MyoRq4c.
[27] Taiwoo Park, Jinwon Lee, Inseok Hwang, Chungkuk Yoo, Lama Nachman, and Junehwa Song. 2011. E-Gesture: A Collaborative Architecture for Energy-efficient Gesture Recognition with Hand-worn Sensor and Mobile Devices. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems (SenSys '11)*. 260–273. https://doi.org/10.1145/2070942.2070969
[28] Lionel Pigou, Sander Dieleman, Pieter-Jan Kindermans, and Benjamin Schrauwen. 2015. *Sign Language Recognition Using Convolutional Neural Networks*. Springer International Publishing, Cham, 572–578. https://doi.org/10.1007/978-3-319-16178-5_40

[29] 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 and Networking (MobiCom '13)*. 27–38. https://doi.org/10.1145/2500423.2500436

[30] Luis Quesada, Gustavo López, and Luis A. Guerrero. 2015. *Sign Language Recognition Using Leap Motion.* Springer International Publishing, Cham, 277–288. https://doi.org/10.1007/978-3-319-26401-1_26

[31] Wenjie Ruan, Quan Z. Sheng, Lei Yang, Tao Gu, Peipei Xu, and Longfei Shangguan. 2016. AudioGest: Enabling Fine-grained Hand Gesture Detection by Decoding Echo Signal. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '16)*. 474–485. https://doi.org/10.1145/2971648.2971736

[32] C. Savur and F. Sahin. 2015. Real-Time American Sign Language Recognition System Using Surface EMG Signal. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. 497–502. https://doi.org/10.1109/ICMLA.2015.212

[33] Jiacheng Shang and Jie Wu. 2017. A Robust Sign Language Recognition System with Multiple Wi-Fi Devices. In *Proceedings of the Workshop on Mobility in the Evolving Internet Architecture (MobiArch '17)*. 19–24. https://doi.org/10.1145/3097620.3097624

[34] SignAll. 2017. SignAll Prototype. (2017). Retrieved July 23, 2017 from http://www.signall.us.

[35] StartASL. 2017. Basic Words in Sign Language. (2017). Retrieved July 14, 2017 from https://www.startasl.com/basic-words-in-sign-language.html.

[36] Chao Sun, Tianzhu Zhang, and Changsheng Xu. 2015. Latent Support Vector Machine Modeling for Sign Language Recognition with Kinect. *ACM Trans. Intell. Syst. Technol.* 6, 2, Article 20 (March 2015), 20 pages. https://doi.org/10.1145/2629481

[37] Li Sun, Souvik Sen, Dimitrios Koutsonikolas, and Kyu-Han Kim. 2015. WiDraw: Enabling Hands-free Drawing in the Air on Commodity WiFi Devices. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom '15)*. 77–89. https://doi.org/10.1145/2789168.2790129

[38] Sheng Tan and Jie Yang. 2016. WiFinger: Leveraging Commodity WiFi for Fine-grained Finger Gesture Recognition. In *Proceedings of the 17th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '16)*. 201–210. https://doi.org/10.1145/2942358.2942393

[39] David Tse and Pramod Viswanath. 2005. *Fundamentals of Wireless Communication.* Cambridge University Press.

[40] Aditya Virmani and Muhammad Shahzad. 2017. Position and Orientation Agnostic Gesture Recognition Using WiFi. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '17)*. 252–264. https://doi.org/10.1145/3081333.3081340

[41] Jue Wang, Deepak Vasisht, and Dina Katabi. 2014. RF-IDraw: Virtual Touch Screen in the Air Using RF Signals. In *Proceedings of the 2014 ACM Conference on SIGCOMM (SIGCOMM '14)*. 235–246. https://doi.org/10.1145/2619239.2626330

[42] Wei Wang, Alex X. Liu, and Ke Sun. 2016. Device-free Gesture Tracking Using Acoustic Signals. In *Proceedings of the 22Nd Annual International Conference on Mobile Computing and Networking (MobiCom '16)*. 82–94. https://doi.org/10.1145/2973750.2973764

[43] Hikaru Watanabe, Masahiro Mochizuki, Kazuya Murao, and Nobuhiko Nishio. 2016. A Recognition Method for Continuous Gestures with an Accelerometer. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct (UbiComp '16)*. 813–822. https://doi.org/10.1145/2968219.2968291

[44] Hongyi Wen, Julian Ramos Rojas, and Anind K. Dey. 2016. Serendipity: Finger Gesture Recognition Using an Off-the-Shelf Smartwatch. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. 3847–3851. https://doi.org/10.1145/2858036.2858466

[45] Chao Xu, Parth H. Pathak, and Prasant Mohapatra. 2015. Finger-writing with Smartwatch: A Case for Finger and Hand Gesture Recognition Using Smartwatch. In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications (HotMobile '15)*. 9–14. https://doi.org/10.1145/2699343.2699350

[46] Sangki Yun, Yi-Chao Chen, Huihuang Zheng, Lili Qiu, and Wenguang Mao. 2017. Strata: Fine-Grained Acoustic-based Device-Free Tracking. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '17)*. 15–28. https://doi.org/10.1145/3081333.3081356

[47] Zahoor Zafrulla, Helene Brashear, Thad Starner, Harley Hamilton, and Peter Presti. 2011. American Sign Language Recognition with the Kinect. In *Proceedings of the 13th International Conference on Multimodal Interfaces (ICMI '11)*. 279–286. https://doi.org/10.1145/2070481.2070532

[48] Ouyang Zhang and Kannan Srinivasan. 2016. Mudra: User-friendly Fine-grained Gesture Recognition Using WiFi Signals. In *Proceedings of the 12th International on Conference on Emerging Networking EXperiments and Technologies (CoNEXT '16)*. 83–96. https://doi.org/10.1145/2999572.2999582

[49] 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 (UIST '14)*. 527–534. https://doi.org/10.1145/2642918.2647380

Received August 2017; revised November 2017; accepted January 2018