Interacting with Soli: Exploring Fine-Grained Dynamic Gesture Recognition in the Radio-Frequency Spectrum

Saiwen Wang[1]†, **Jie Song**[1]†, **Jaime Lien**[2], **Ivan Poupyrev**[2], **Otmar Hilliges**[1]
[1]ETH Zurich, [2]Google ATAP
{jsong|otmar.hilliges}@inf.ethz.ch, sawang@student.ethz.ch, {jaimelien|ipoupyrev}@google.com

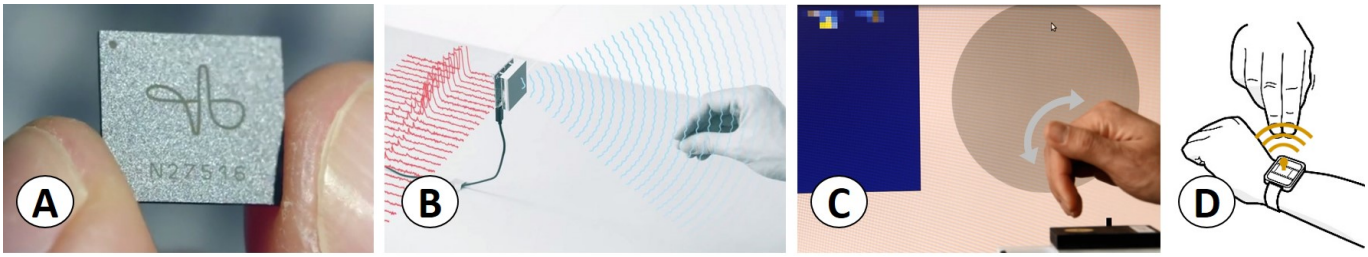

Figure 1. We explore interactive possibilities enabled by Google's project Soli (A), a solid-state short-range radar, capturing energy reflected of hands and other objects (B). The signal is unique in that it resolves motion in the millimeter range but does not directly capture shape (C). We propose a novel gesture recognition algorithm specifically designed to recognize subtle, low-effort gestures based on the Soli signal.

ABSTRACT

This paper proposes a novel machine learning architecture, specifically designed for radio-frequency based gesture recognition. We focus on high-frequency (60 GHz), short-range radar based sensing, in particular Google's Soli sensor. The signal has unique properties such as resolving motion at a very fine level and allowing for segmentation in range and velocity spaces rather than image space. This enables recognition of new types of inputs but poses significant difficulties for the design of input recognition algorithms. The proposed algorithm is capable of detecting a rich set of dynamic gestures and can resolve small motions of fingers in fine detail. Our technique is based on an end-to-end trained combination of deep convolutional and recurrent neural networks. The algorithm achieves high recognition rates (avg 87%) on a challenging set of 11 dynamic gestures and generalizes well across 10 users. The proposed model runs on commodity hardware at 140 Hz (CPU only).

Author Keywords

gesture recognition; wearables; deep learning; radar sensing

ACM Classification Keywords

H.5.2 User Interfaces; I.3.6 Methodology and Techniques;

INTRODUCTION

As computing moves increasingly beyond the desktop the HCI community has researched a multitude of alternative input strategies. In particular interaction via whole body or hand gestures (often called natural user interfaces) has seen tremendous interest in recent years. In the context of mobile and wearable computing this remains a challenging problem because instrumenting the user or the environment does not scale well: camera based solutions which have reached high levels of accuracy in stationary settings are not well suited for ubiquitous mobile gesture interaction due to the need for direct line of sight and issues with (self-)occlusion.

Non-vision based sensing such as electronic field sensing and radio frequency sensing has been proposed to alleviate some of these issues but so far suffered from low spatial resolution. However, emerging high-frequency ultra-wideband chips [12] are the basis for digital millimeter-wave radars which promise to combine high accuracy at short range with low-power consumption and compact physical form-factor. One instance of such sensors is Google's project Soli [23], a new sensor developed for interactive input recognition on mobile and wearable devices.

Millimeter-wave radar has the potential to serve as basis for mobile gesture recognition by overcoming many issues in vision based and low-frequency RF sensing. However, it brings a number of new challenges for HCI and gesture recognition research: sensing in the electro-magnetic spectrum eschews spatial information for temporal resolution. Capturing a superposition of reflected energy from multiple parts of the hand such as the palm or fingertips, the signal is therefore *not directly suitable* to reconstruct the spatial structure or the shape of objects in front of the sensor. However, the signal *does capture motion* even of very small magnitude and it is possible to discriminate very subtle and precise hand motions and gestures. Embracing this challenge we propose a novel

†The first two authors contributed equally to this work.

UIST 2016, October 16–19, 2016, Tokyo, Japan.

http://dx.doi.org/10.1145/2984511.2984565

deep learning based gesture recognition approach specifically designed for the recognition of dynamic gestures with millimeter wavelength RF signals.

Our main contributions are: (1) a novel end-to-end trained stack of convolutional and recurrent neural networks (CNN/RNN) for RF signal based *dynamic* gesture recognition. (2) The algorithm runs in real-time on off-the-shelf hardware and shows the potential to robustly recognize even challenging micro-interactions. (3) Furthermore, we contribute an in-depth analysis of sensor signal properties and highlight inherent issues in traditional frame-level approaches. Finally, the most important design choices and parameters of the recognition algorithm are discussed.

We conclude with an outlook onto exciting opportunities for research in the context of short range radar sensing, including end-to-end gesture spotting and gesture recognition as well as continuous 3D tracking.

RELATED WORK

Input recognition is an important area in HCI research, the need for alternative input paradigms has only grown with the explosive proliferation of mobile computing. We focus our literature review on (1) various suitable sensing modalities and (2) gesture recognition algorithms.

Sensing modalities:

A number of camera based solutions exist, with early work focusing on 2D RGB cameras [7]. In-air gestures in on-the-go scenarios have been shown using either wrist-worn cameras [20] or utilizing the mobile devices built-in camera [34]. More recent methods have enabled fine-grained 3D hand-pose estimation in real-time from depth-images (e.g., [19, 31]). However, size and the requirement for direct line of sight mostly restrict the applicability of these approaches for ultra mobile settings. In the light of these challenges researchers have explored several alternative sensing modalities. Including simple IR proximity sensors facing outwards [2], or upwards [21], and magnetic field sensing for tracking of rigid motion around a device [18]. Electric field sensing has been used to track a single fingertip in 3D above a mobile device [8], or to recognize a small number of full-body gestures [4]. Furthermore, the Doppler shift can be exploited for audio-based sensing [11]. Finally, capacitive effects can be used for input recognition (e.g., [28]).

Most closely related to our work are approaches that leverage radio frequency waves to detect motion and gestural input. Examples include sensing via disturbances of GSM [39] signals, picking-up electromagnetic interference in LCDs [3] or piggybacking onto existing WiFi signals [26]. We refer to [27] for an overview. To main difference to our work lies in the nature of the signal. Existing approaches operate in relatively low frequency bands (less than 5 GHz), inherently limiting spatial resolution, whereas we use high-frequency radar with a central frequency of 60 GHz, which allows for more fine grained gesture sensing.

Input recognition algorithms:

Due to the complexity of human motion and difficulty to accurately sense it, many gesture recognition algorithms are now based on some form of machine learning model. Traditional architectures require features exctracted from low-level data. Typical choices for camera based sensing include spatio-temporal features (e.g., [32, 34, 35]), capturing shape and motion cues. Similarly, non-camera based sensing often relies on features designed according to domain knowledge [11, 26]. Such feature vectors are then classified into different gesture classes via appropriate ML models (often SVMs). Identifying features with relevant information content is known to be a time consuming and fragile process [39].

In part motivated by difficulties in finding good hand-crafted features, there is now a growing trend toward feature representations learning via deep neutral networks. Such approaches have been successfully applied to various tasks in video based action recognition [33] and speech recognition [15]. Similarly [24] uses CNNs for sign language recognition based on combined color and depth data. Finally, significant improvements in human activity recognition based on accelerometers or gyroscopes have been demonstrated [25] by leveraging similar architectures.

A further challenging aspect of gesture recognition is that of modelling the dynamics a motion sequence. Hidden Markov Models have been applied to several gesture recognition and HCI tasks [22, 37]. Other state space based models include dynamic time warping [5] and finite state machines [16]. Recently, Recurrent Neural Networks (RNN), especially augmented with *Long short-term memory* (LSTM) cells has been shown to capture sequential information in several domains including speech- [15], video based action [6] and activity recognition [25].

A simple gesture recognition approach alongside the low-level hardware details of the Soli sensor are discussed in [23]. The recognition approach only detects four gestures and only from a single user. In this paper, we propose a joint model to learn feature representations and the dynamic patterns in high frequency radar signals. The model is trained in an end-to-end manner, can recognize larger gesture sets with high accuracy and it is robust across users and sessions. Finally, we make the pre-trained model publicly available for others to be able to extend our work*.

RADAR-BASED GESTURE RECOGNITION

We build upon Google's Project Soli [23], a purpose-build sensor for micro-interactions in mobile and wearable computing. We are specifically motivated by the ability to sense subtle, fast, precise, unobtrusive and low-effort gestures, involving primarily small muscle groups to prevent fatigue over time and natural motion. Such gestures would provide a good trade-off between precision of interaction and effort [38]. For example, sliding the thumb over the index finger for 2D relative input which could be mapped to a cursor or used to navigate a hierarchical menu. Other interesting interactions may include diminutive flicks of individual fingers or dynamic trajectories of fingertips (without moving the wrist).

However, while Soli and similar sensors hold the potential to bring such micro-interactions to mainstream mobile devices,

*`http://ait.ethz.ch/ds/`

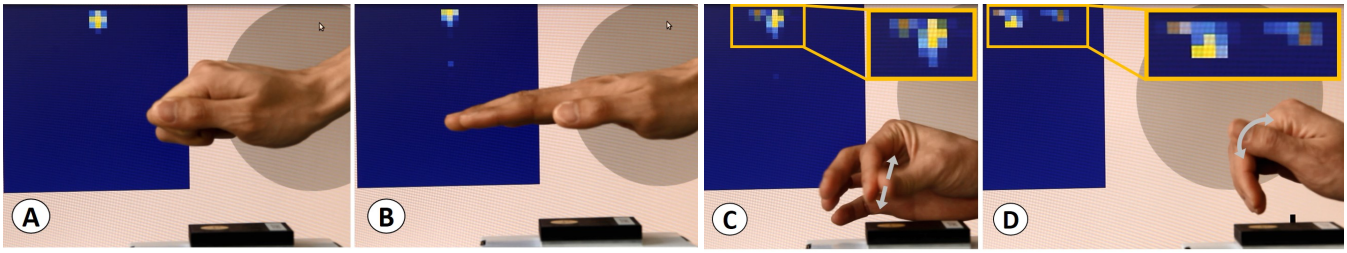

Figure 2. Signal properties. Pixel intensity corresponds to reflected energy; horizontal axis is velocity; vertical axis is range. (A+B) Sensor produces almost identical response for static objects even of distinct shape. (C+D) In contrast, the sensor resolves even minute motion with high resolution.

there are many challenging problems that need to be overcome due to the nature of the signal. Before detailing our approach we now discuss the most salient properties of the signal and its implications for input recognition.

Hardware

The Soli sensor [23] is a solid-state millimeter-wave radar for mobile gesture recognition. Classic radar approaches rely on high spatial resolution to discern several rigidly moving targets (e.g., planes). In contrast, Soli uses a sensing approach that prioritizes high temporal resolution to detect subtle, non-rigid motion. Soli utilizes a single broad antenna beam to illuminate the entire hand as modulated pulses are transmitted at very high repetition rates (between 1-10 kHz).

Signal

The raw received signal, consisting of a superposition of reflections from scattering centers within the radar's antenna beam, is then processed into multiple abstract signal representations. The high temporal resolution enables a combination of fast time and slow time processing to map scattering center reflections into interpretable dimensions (for details see [23]).

Figure 2 illustrates the Range-Doppler images (RDI) attained by mapping received energy into a two-dimensional space of radial distance (or range) and velocity. Each blob's pixel intensity corresponds to the reflected energy received from each scattering center. The pixel positions correspond to that scattering center's distance and velocity. Multiple scattering centers are then resolvable in either range or velocity within constraints imposed by the radar point target response, a function of the transmission parameters and low-level signal processing. This procedure results in a time-varying two-dimensional image in which the trajectories of the blobs over time indicate the gesture motion pattern (cf Figure 2, C+D).

Implications for input recognition

We now turn our attention to gesture recognition. Here many methods are based on images [20, 31, 35] and rely on spatial information. As shown in Figure 2 (A+B), short-range radar data does not directly contain information about shape and hence many existing algorithms are not applicable.

However, there are communalities to other non-camera methods, particularly those that rely on the Doppler effect (e.g., [11, 39]), which traditionally have been limited to coarse gestures and small gesture sets. These approaches can be summarized as a 3-step process consisting of signal transformations (e.g., Fourier transform) followed by feature extraction and a frame-level classification step. In [23] such an approach is used, capable of discriminating four gestures from a single user. We have conducted an extensive set of comparisons with a variety of ML algorithms on a 10 gesture dataset collected from 5 users. Approaches included random forests using per-frame features (acc: 26.8%), RFs with temporal representation based on RDIs (acc: 30%) and HMMs (acc: 35%). While these poor results are indicative of the issues at hand, we note that it is impossible to exhaustively explore all possible algorithms and parameters and hence negative results can never be fully conclusive.

Figure 2 illustrates the issue at the heart of this problem – the sensor does not resolve shape of objects with high *spatial* resolution but instead provides high *temporal* resolution, capturing primarily *changes* in hand-pose. In Figure 2 (C), the user rubs the fingertips together and slides the thumb over the index finger in Figure 2 (D). Both interactions would be hard to segment and track in the spatial domain but are clearly discernable in the Range-Doppler space (see insets). In other words the issue is one of frame-level classification, which primarily leverages instantaneous properties (e.g., shape), versus approaches that can effectively model dynamics.

METHOD

We propose a deep-learning architecture designed for gesture recognition with high-frequency radar. While our implementation is specific to the Soli sensor, generalizing the approach to other high-frequency RF signals is straightforward.

Deep Learning with Neural Networks

Our architecture, schematically shown in Figure 3, combines the steps of *representation learning* and *dynamic modelling* into a single end-to-end model. Foregoing attempts to reconstruct shape from the signal, we directly use the final gesture recognition as quantity to optimize for during training. We show experimentally that our approach outperforms a variety of alternative designs (see discussion of results).

Representation Learning

Gesturing above the sensor results in a sequence of aligned Range-Doppler images which can be thought of as a stream of the dynamics of changing hand configurations (cf. Figure 3, top to bottom). The first step in any machine learning pipeline is to extract features from the data. Traditionally this has been done manually (e.g., [11, 30, 26]) but recently CNNs have been successful in a variety of challenging tasks (e.g., image classification [14]) in learning features automatically. While

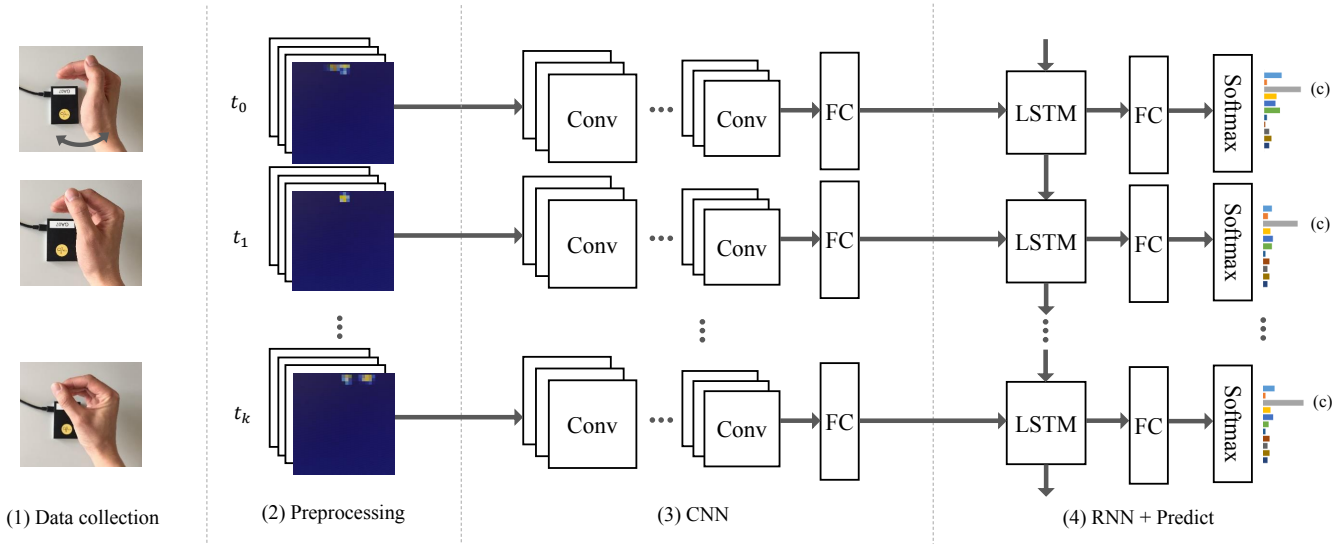

Figure 3. Gesture recognition pipeline. (1) Data produced by the sensor when sliding index finger over thumb. (2) Preprocessing and stacking of frames. (3) Convolutional Neural Networks. (4) Recurrent Neural Networks with per-frame predictions.

not encoding shape, the RDIs still contains interpretable information about the motion of reflection centers, and CNNs can extract useful intermediate representations.

More specifically the network learns a feature representation function $f(I_t, W)$ that maps inputs I_t (i.e., Range-Doppler images) to outputs x_t, where W contains the weights of the network. During learning we use the classification error of the overall pipeline as optimization objective. Designing CNN architectures is a complex task involving many hyper-parameters such as the number of layers and neurons, activation functions and filter sizes. In the experiments section we report on different CNN variants. Most saliently, we compare a network adapted from computer vision [33] to a network that we designed specifically for the Soli data. Please note that neither CNN variant alone provided enough discriminative power for gesture recognition.

Dynamic Gesture Modelling

A number of algorithms exist to model dynamic processes, including Bayesian networks and HMMs specifically. HMMs have been used for gesture recognition and other HCI tasks (e.g., [37, 22]). Recently they have been outperformed by recurrent neural networks (RNNs) [9], augmented with *Long short-term memory* (LSTM) [10] cells. Since CNNs alone can not exploit the temporal information embedded in our data we use an LSTM RNN for the modelling of dynamics.

Recurrent Neural Networks differ from feedforward networks in that they contain feedback loops, encoding contextual information of a temporal sequence. Given an input sequence $\mathbf{x} = (x_1, ..., x_T)$, where in our case the x_t is the feature vector extracted by the CNN at time t, the hidden states of a recurrent layer $\mathbf{h} = (h_1, ..., h_T)$ and the outputs $\mathbf{y} = (y_1, ..., y_T)$ can be attained:

$$\begin{aligned} h_t &= H(W_{ih}x_t + W_{hh}h_{t-1} + b_h) \\ y_t &= W_{ho}h_t + b_o \end{aligned} \tag{1}$$

where Ws are weight matrices connecting input, hidden and output layers, b are bias vectors and H is the hidden layer's activation function. Crucially, the hidden states h_t are passed on from timestep to timestep while the outputs y_t are fed to a softmax layer, providing per-frame gesture probabilities.

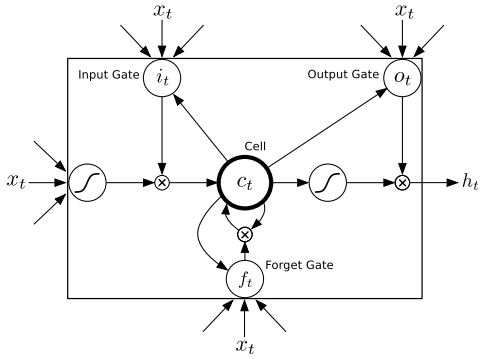

Figure 4. LSTM input, output, cell and gates connections. Showing the relationship between gates connections and memory cell location.

When learning over long sequences, standard RNNs can suffer from numerical instabilities known as the vanishing or exploding gradient problem. To alleviate this issue LSTMs use memory cells to store, modify, and access internal state via special gates, allowing for better modelling of long-range temporal connections. For each unit, the relation between input, internal state and output is formulated as follows:

$$\begin{aligned} i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \\ f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \\ c_t &= f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\ o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \\ h_t &= o_t \tanh(c_t) \end{aligned} \tag{2}$$

where σ is the logistic sigmoid function, and the input gate, forget gate output gate and cell activation are respectively represented by i, f, o and c (see Figure 4 for an illustration).

End-to-end training

Figure 3 shows how the CNN and LSTM interact at evaluation time. Importantly, we train these two models jointly in an end-to-end manner rather than individually. For training we pass each Range-Doppler image belonging to a gesture sequence through the CNN for feature extraction. The outputs are then passed into the LSTM units. For each time step and corresponding y_t a softmax layer in the RNN predicts a gesture label for which we compute a loss value and then use back-propagation to compute the gradient of the network's weights W. Gradients and loss values are then summed over time and finally the network's parameters are optimized, minimizing the combined loss similar to [36].

SYSTEM EVALUATION

The goal of our work is to enable recognition of dynamic gestures of the type illustrated in Figure 5. In this section, we provide an in-depth experimental analysis of the impact of different network architecture design choices. We demonstrate that our proposed method can discriminate a large set of gestures with an average accuracy of 87.17% and can even discriminate gestures with identical trajectories based on gesture dynamics alone. A personalized classifier achieves an accuracy of 88% which can be further improved to 94.5% using sequence pooling, albeit at the cost of additional latency and requiring accurate gesture spotting.

Gesture set

Our final gesture set is illustrated in Figure 5. While we do not claim this to be the definitive gesture set, it serves the purpose of exploring both Soli and the algorithms potential for input recognition. The gestures were selected via a lengthy process including literature review, design workshops, focus groups and Wizard-of-Oz user evaluations.

Design principles

From the above process we distilled the following three guiding principles: (1) *Micro-gestures* we prefer interactions involving small amounts of motion and those that are performed primarily by muscles driving the fingers and articulating the wrist, rather than those involving larger muscle groups to avoid fatigue over time. (2) *Proprioception* we want to support proprioception and physical (self-)support in gestures. For example, a finger resting on another while sliding over the digits to simulate a pointing device (cf. Figure 5, c+i) or fingers touching each other (Figure 5, a+d). (3) *Dynamic gestures* given the properties of the signal most gestures are dynamic and involve motion – typically of small magnitude.

We initially designed a set of 15 candidate gestures included several gestures to explicitly evaluate the discriminative power of the signal and the ML-architecture. In particular, gestures hard or impossible to recognize with camera based sensing (e.g., the LeapMotion sensor), such as *Finger-Rub*, due to self-occlusion, small motion and contact between multiple hand parts. We recruited subjects from our institution for an informal pre-study. Based on this feedback, we decided to remove four initial gestures where users indicated that they were hard to perform and memorize. Interestingly these were mostly trajectory based gestures (i.e., in-air writing) and users often performed them by moving the entire arm, rather than just using individual fingers. Furthermore, we refined the remaining 11 gestures (cf. Figure 5).

Training and test data

Training deep neural networks requires large amounts of training data which has to contain sufficient variation in terms of gesture execution. We asked 10 subjects, again recruited from our institution, to perform the 11 gestures, receiving only minimal instruction on how-to perform them (resulting in large variability). Instances of each gesture were annotated with a class label and only clear outliers were removed from the dataset. We recorded raw Range-Doppler images at 40Hz and captured each gesture 25 times from all 11 subjects, over 10 sessions resulting in $11 \times 25 \times 10 = 2750$ sequences, each a couple of seconds long. This data set is used in most of our experiments. To evaluate cross-session performance and to explore personalized gesture recognition (i.e., classifier trained on a per-user basis), we recorded an additional $(11 \times 50 \times 5 = 2750)$ sequences from a single user.

For training and evaluation, a random shuffle 50%-50% split is used unless stated differently. For cross validation we use a standard k-fold leave-one-subject/session-out approach.

Input Preprocessing

Before feeding the data to our model we perform background removal (using a per-pixel Gaussian model) and signal normalization. The latter is necessary since the range of the Soli sensor is currently limited to 30 cm, causing large variance in values recorded for reflection centers close to the sensor to those far away. We scale the data logarithmically and perform a max-min truncation to reduce the influence of outliers. The input to our models is formed by stacking consecutive Range-Doppler frames (shown visually in Figure 3, pt. (2)). Stacked frames explicitly capture temporal changes (cf. Figure 6).

Experimental conditions

Here we contrast different variants of network architectures. Furthermore, we compare the proposed end-to-end architecture in terms of accuracy, memory consumption and runtime performance. In particular, we perform an in-depth analysis of the following architectures:

- **Frame-level classification** As baseline we use a standalone CNN for representation learning only. Based on the learned features we perform traditional frame-level gesture classification. This allows us to assess the discriminative power of the raw data and its transformations. Particularly, we compare the performance of an adapted network from computer vision community [33], to a network design specifically with our radar data. To further analyze the property of radar data representation learned with CNN, we also compare the performance difference when removing pooling layers of a CNN (see table 1 column 2).

- **Standalone RNN** To evaluate the impact of temporal coherence we train a standalone RNN that takes features ex-

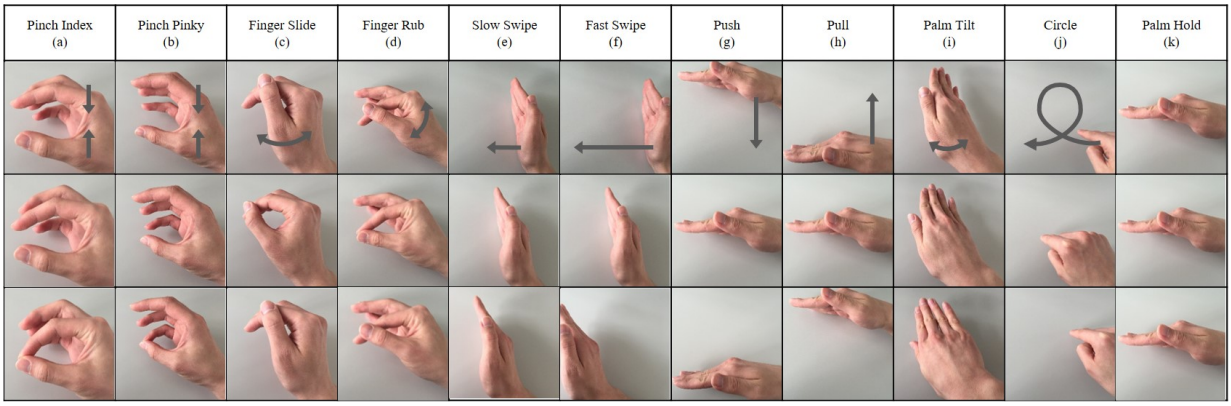

Figure 5. Full gesture set used for experiments. Rows illustrate temporal sequence (top to bottom). Arrows indicate motion trajectories.

tracted by a CNN as input. Importantly both models are trained independently.

- **End-to-end model** we use the model shown in Figure 3 where CNN+RNN networks are trained jointly based on temporally coherent prediction loss.

Implementation Details

We implemented and evaluated different CNN architectures (summarized in Table 1). The deep network architecture has been adapted from [33], where it is used for video based action recognition. This network uses 8 layers, large convolutional kernels, large max-pooling windows and also uses more nodes in the fully-connected layers. For compatibility with this network, raw RDI data is resized to a 224×224 resolution. The shallow network has been designed specifically for the RDI data. We use three convolutional layers with 2×2 and 3×3 filters. For the convolution layers we use no padding and stride one, whereas the max-pooling layers use the same stride as the filter size in the convolutional layer below. To analyze the necessity of pooling layers in CNN, we further compare to a network with the only change of removing the pooling layers. We use the *Rectified-Linear* (ReLU) activation function and local response normalization is applied after each layer. Dropouts of 0.9 and 0.8 are applied in the first two fully connected layers respectively. The network is trained by mini-batch stochastic gradient descent with momentum of 0.9 and a batch size of 64. The initial training rate is set to 10^{-3} and decreased to 10^{-4}, 10^{-5} and 10^{-6} at iteration 5K, 8K and 11K, stopping after 15K iterations.

For experiments using sequences as input (standalone RNNs and end-to-end models), the sequences are either subsampled or completed by inserting frames, resulting in sequences with uniform length of 40 frames. The alignment of sequence makes batch training and evaluation possible, which smooths the optimization process and at the same time accelerates the training process by exploring the parallelism of GPUs. The network uses dropouts after the output of each LSTM cell with ratio of 0.5 and is trained in batches of 16 sequences each. The sequences are randomly shuffled and the frame order is maintained (this is essential for the RNN). Learning rate starts at 10^{-3} and is decreased to a tenth every 20 epochs (an epoch is defined as one full pass of training data). The training stops after 50 epochs. As for the end-to-end models, the last fully connected layer of CNN is connected directly to the input of LSTM cell. Dropouts of 0.4 are applied to the last two convolutional layers and dropouts of 0.5 are applied to the fully connected layers. The whole network is trained end-to-end and parameters in both networks are optimized jointly, similar to the training of the standalone RNN.

Realtime considerations

The two CNN network design choices discussed above also impact training and runtime behavior. While a deeper net should display better accuracy, this obviously impacts memory consumption, training and run-time efficiency negatively.

The deeper model (CNN only) consumes $52\times$ more memory than our proposed model (1.2 GB vs 23 MB), consumes $5.4\times$ more GPU memory at runtime [†] (1093 MB vs 204 MB) and takes $4\times$ more time per frame prediction than the shallow model (80 Hz vs 395 Hz). The final end-to-end model proposed in this paper (CNN+RNN) has a model size of 689 MB, consumes 265 MB GPU memory at runtime and predicts frames at a rate of 150 Hz. Crucially this model can be evaluated on commodity hardware whereas the deeper architecture has to be run on a server with high-end GPUs. In contrast, our model can even be evaluated by a pure CPU implementation running at 140 Hz on an Intel® Core™i7 CPU.

Results

Table 2 summarizes the overall classification accuracies of the various network architectures. First we unpack the impact of frame-to-frame feature learning on classification.

Frame-level classification

Overall we compared three variants of the standalone CNN, summarized in the first three rows of Table 2. While our results indicate that using deeper networks improves accuracy (avg. 48%) over either of the shallower CNN variants (avg. 41%), the improvement of 7% is not sufficient for usable gesture recognition. Unpacking this further we can see that some gestures, namely the *Fast Swipe* (97.6%) and *Hold Palm* (83.2%) are relatively easy to distinguish. This intuitively makes sense as these either contain no motion or fast

[†] NVIDIA GeForce GTX TITAN X GPU with 12GB of memory.

Layers	*CNN shallow w/ pooling*	*CNN shallow w/o pooling*	*CNN deep*	*RNN*	*End-to-end*
1	**conv1** $3 \times 3 \times 32$ - **pool1** 2×2	**conv1** $3 \times 3 \times 32$	**conv1** $7 \times 7 \times 96$ - **pool1** 3×3	**fc1** 512	**conv1** $3 * 3 * 32$
2	**conv2** $3 \times 3 \times 64$ - **pool2** 2×2	**conv1** $3 \times 3 \times 64$	**conv2** $5 \times 5 \times 256$ - **pool2** 3×3	**lstm2** 512	**conv2** $3 * 3 * 64$
3	**conv3** $3 \times 3 \times 128$ - **pool3** 2×2	**conv1** $3 \times 3 \times 128$	**conv3** $3 \times 3 \times 512$	**fc3** - **softmax** 11	**conv3** $3 * 3 * 128$
4	**fc4** 512	**fc4** 512	**conv4** $3 \times 3 \times 512$	-	**fc4** 512
5	**fc5** 512	**fc5** 512	**conv5** $3 \times 3 \times 512$ - **pool5** 3×3	-	**fc5** 512
6	**fc6** - **softmax** 11	**fc6** - **softmax** 11	**fc6** 4096	-	**lstm6** 512
7	-	-	**fc7** 2048	-	**fc7** - **softmax** 11
8	-	-	**fc8** - **softmax** 11	-	-

Table 1. Comparison of network architectures. Network architectures used in our experiments. For standalone CNN, we compare two groups of networks with different depth: a shallow network optimized for radar data and a deeper network. For the shallow network, an extra network without pooling layers is listed for comparison. For the end-to-end architecture, we only list the configuration without pooling layers, outperforming those with pooling layers.

Network Architecture	*Avg. Acc.*	 *(a)*	 *(b)*	 *(c)*	 *(d)*	 *(e)*	 *(f)*	 *(g)*	 *(h)*	 *(i)*	 *(j)*	 *(k)*
CNN Shallow	40.80%	18.00%	36.00%	3.60%	47.60%	39.60%	96.80%	27.60%	24.00%	51.20%	20.00%	84.40%
CNN Shallow (1)	41.13%	20.00%	34.40%	1.20%	46.80%	37.20%	97.60%	29.60%	22.00%	54.00%	26.4%	83.20%
CNN Deep	48.18%	47.60%	34.40%	6.80%	39.60%	43.20%	98.80%	56.00%	31.20%	52.80%	48.40%	71.20%
RNN Shallow FC (2)	77.71%	60.35%	62.25%	38.72%	89.45%	66.77%	92.52%	94.93%	86.89%	91.39%	85.52	86.22
EtE	85.22%	43.99%	77.85%	79.23%	95.32%	81.81%	95.28%	98.18%	87.88%	95.16%	92.27%	90.43
EtE w/o Pooling	**87.17%**	67.72%	71.09%	77.78%	94.48%	84.84%	98.45%	98.63%	88.89%	94.85%	89.56%	92.63%
EtE CV-sub (3)	79.06%	58.71%	67.62%	64.80%	91.82%	72.31%	72.91%	93.40%	89.99%	95.16%	82.80%	80.24%
EtE CV-sub Avg (4)	88.27%	70.80%	76.80%	83.20%	97.20%	80.40%	83.60%	94.80%	100.00%	100.00%	97.20	86.82%
EtE CV-ses (5)	85.75%	56.69%	61.98%	76.43%	96.83%	92.73%	81.38%	98.42%	97.79%	95.33%	96.92%	89.10%
EtE CV-ses Avg (5)	94.15%	79.20%	74.40%	95.60%	100.00%	97.60%	94.80%	100.00%	100.00%	100.00%	100.00%	94.09%

Table 2. Accuracy of each gesture under different configurations. The first six configurations use 50%-50% split for training and evaluation. The latter four use cross-validation. (1) CNN shallow network without pooling layers. (2) FC (last fully connected layer before softmax layer) output of CNN shallow net. (3) Leave-one-subject-out cross-validation on 10 subject with per-frame accuracy. (4) Leave-one-subject-out cross-validation on 10 subject with sequence average-pooled accuracy. (5) Leave-one-session-out cross-validation on single subject with per-frame accuracy. (6) Leave-one-session-out cross-validation on single subject with sequence average-pooled accuracy.

motion of the entire hand as a single rigid-object. These findings are in-line with prior work detecting similar gestures from Doppler data in a frame-to-frame manner [11]. More importantly, the data clearly shows the difficulties of detecting *dynamic* gestures in a frame-level setup, further verifying our assessment of the signal characteristics.

To understand the impact of pooling layers in the CNN, we compare the performance of the shallow CNN to the same configuration but without pooling layer. We observe that removing pooling layers from the CNN does slightly improve the prediction accuracy (cf. Table 2, rows 1+2). Removing pooling layers from the end-to-end configuration yields similar results (cf 5$^{\text{th}}$ and 6$^{\text{th}}$ rows of Table 2).

Unpacking this further we note that gesture *circle*, which relies on absolute scattering location for classification sees drastic performance boost from removing the pooling layer. The gesture pair *fast swipe* and *slow swipe* which have very similar dynamic patterns but differ in the magnitude of the trajectory, encoded as blob positions in the Range-Doppler image sequence. Similarly to the standalone CNN, removing pooling layers yields better classification performance for these two gestures.

Stand-alone RNN with input from CNN

Due to the relatively small differences in power between the two CNN variants and the implications for real-time use, we drop the deeper architecture in further experiments. Row 4 of Table 2 show results from using an RNN on top of an independently trained CNN. The overall per-frame accuracy is drastically improved by introducing a dynamic model (avg. 77.7%) – an improvement of 30% over the best frame-level result, clearly supporting our approach of combining a representation learning step with a dynamic model. However, the model struggles with important gestures containing very subtle motion, especially *Finger Slide*.

To further visualize the effectiveness of using contextual information (in our case LSTM as memory cell) for gesture recognition, we take a pair of gestures *fast swipe* and *circle* as example. Figure 6 illustrates the temporal evolution of the example sequences. Both gestures have very similar spatial blob patterns but the trajectories of the scattering centers are clearly distinguishable. In table 2 we see an accuracy boost for the *circle* gesture. This can be explained by its double loop trajectory which is captured by the RNN's memory cell state. We experimented with stacking multiple RNN layers but this did not yield significant improvements.

End-to-end model with CNN+RNN

Table 2 (row 5) summarizes the results attained by training CNN and RNN jointly. The end-to-end model further improves the average accuracy to 87% and increases the accuracies of gestures that are problematic in the standalone case.

Looking at the probability evolution over the duration of sequences, plotted in Figure 7 further highlights the importance

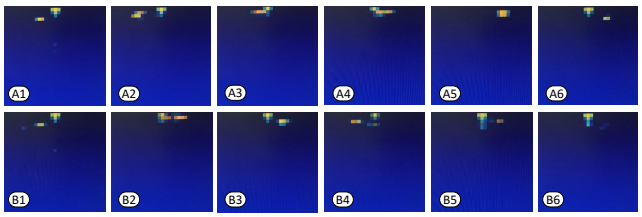

Figure 6. Range-Doppler temporal evolution of two seperate gestures. Two gestures (A) *fast swipe* and (B) *circle* have very similar spatial blob patterns, but very different temporal trajectories

of dynamic gesture modelling. Consistent with the frame-level CNN performance, some gestures (e.g., *Fast Swipe*) can be recognized almost instantaneously, whereas other gestures clearly benefit from modelling the temporal effects. For example, *Palm Tilt* and *Pull* can be detected with high confidence after short period (roughly 10 frames). The gestures *Finger Slide* and *Draw Circle* carry the most information in their trajectories and hence prediction confidence rises with time. The most difficult gesture turned out to be *Pinch Index*, explicable by the small amount of motion and large parts of the fingers being blocked by the palm.

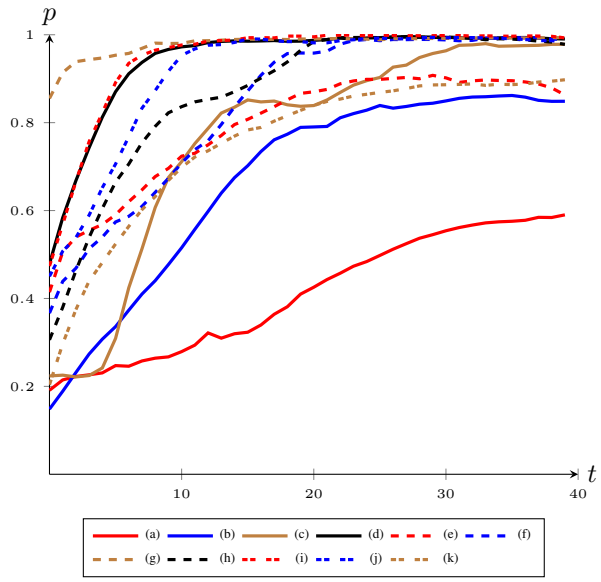

Figure 7. Temporal evolution of gesture probabilities over sequence progress. Results from end-to-end model and 11 gestures (see Figure 5).

In addition to the above half-training-half-test evaluation of the classifier we also perform a leave-one-subject-out cross validation over all 10 participants. This experiment is a good predictor of real-world performance since the test data is now entirely unseen. The per-frame accuracy as summarized in the confusion matrix (Table 3) and remains high with an average of 79%. The two *Pinch* and the *Finger Slide* gestures are the most challenging. Again this is very intuitive as these gestures allow for high individual variability. As for gestures like *Finger Rub* and *Push*, the recognition performance remains consistent with the half-training-half-test setting due to low variation between different users.

Extented Experiements

Here we briefly summarize a number of experiments that we think are most promising alleys for future work to attain a real-world system that could be integrated into wearables.

Personalized dataset

One way of improving the real-world accuracy is to personalize the classifier, as wearable devices are usually only worn by a single user. To verify this assumption we use a personalized dataset collected from a single user. Evaluating accuracy across sessions (to simulate real-world user scenarios the network is trained once, and evaluated at runtime with different sessions) improves the recognition rate to 85.75%, which is comparable to accuracies reported in the literature for *static* gestures based on other modalities. For example, several mobile systems report accuracies in the 80% range (87.6% [13], 85% [29]).

GT	Prediction										
	(a)	*(b)*	*(c)*	*(d)*	*(e)*	*(f)*	*(g)*	*(h)*	*(i)*	*(j)*	*(k)*
(a)	**59%**	17%	12%	0%	2%	1%	0%	0%	0%	0%	9%
(b)	16%	**68%**	6%	4%	1%	2%	0%	0%	0%	0%	3%
(c)	14%	9%	**65%**	1%	5%	0%	0%	0%	0%	0%	6%
(d)	0%	3%	0%	**92%**	0%	1%	0%	0%	3%	1%	0%
(e)	4%	1%	6%	2%	**72%**	5%	0%	0%	2%	6%	2%
(f)	1%	0%	0%	0%	0%	**73%**	10%	0%	1%	9%	6%
(g)	0%	0%	0%	0%	0%	2%	**93%**	4%	0%	1%	0%
(h)	2%	3%	1%	1%	0%	0%	1%	**90%**	0%	0%	2%
(i)	1%	1%	1%	0%	1%	0%	0%	0%	**95%**	0%	0%
(j)	0%	0%	0%	0%	5%	8%	1%	0%	3%	**83%**	0%
(k)	13%	2%	2%	0%	2%	1%	0%	0%	0%	0%	**80%**

Table 3. Confusion matrix of end-to-end model from leave-one-subject-out cross-validation. Darker cell-shade indicates higher accuracy.

Sequence Pooling

In contrast to per-frame prediction, a sequence based average-pooled prediction produces even more robust results due to temporal filtering. We sum up the softmax-normalized activation at each time step of a sequence emitted by the RNN, and use the averaged activation for gesture prediction. This simple trick uses the full information of a sequence. Rows 8 and 10 of Table 2 compare pooled and averaged (across the entire sequence) results to per-frame predictions. Not surprisingly, a significant accuracy boost can be observed, reaching 88% for the non-personalized classifier and 94.5% for the personalized classifier. Especially for gestures with strong temporal aspects such as *Pull* and *Palm Tilt* perfect predictions can be achieved even on unseen test data. Using this technique would result in a very robust real-world classifier but adds a latency penalty for real-time recognition since pooling can only be performed after the gesture has been completed. At the same time this approach requires a sophisticated gesture spotting algorithm (i.e., gesture segmentation).

Gesture spotting

So far we have segmented gestures with simple heuristics based on experimentally derived thresholds. In this sense a final but important step towards a real-world implementation is that of gesture spotting (i.e., detecting the presence of a hand

Config/Gestures	*Accuracy*	*Gesture 1*	*Gesture 2*
EtE (1)	80.34%	84.50%	75.90%
EtE (2)	93.88%	95.50%	92.15%
EtE (3)	78.91%	97.54%	59.03%
CNN Presence	100.00%	100.00%	100.00%

Table 4. Accuracy of pairwise gestures under different configurations. (1) Pairwise micro gestures with *Pinch Index* as gesture 1 and *Pinch Pinky* as gesture 2 using end-to-end model. (2) Pairwise velocity variance gestures with *Swipe Slow* as gesture 1 and *Swipe Fast* as gesture 2 using end-to-end model. (3) Pairwise directional variance gestures with *Push* as gesture 1 and *Pull* as gesture 2 using end-to-end model.

versus random motion). To this end we conducted preliminary experiments deploying the shallow CNN trained specifically to discriminate the user's hand from random signal. In our experiments we achieved close to 100% accuracy (see Table 4) however this is based on a stationary sensor whereas in a mobile scenario the signal would be much noisier. However, our results indicate that it may be possible to differentiate gesturing from random motion.

Pairwise Gestures Recognition

The gesture set (in figure 5) are designed intentionally with some comparable pairs, namely the two *pinch* gestures, *swipe* gestures and *push* & *pull* gestures.

The two *pinch* gestures emphasize the radar's capability of distinguishing micro gestures with very small differences. In *Pinch index* the index finger is further away than in *pinch pinky*, while the two gestures are otherwise very similar. When classifying the entire gesture set (cf. Table 2) these two gestures perform relatively poorly. However, when classifying only into pairwise gestures (see 1st row of Table 4), these two gestures can be distinguished well.

Fast swipe and *slow swipe* are two similar gestures that differ only by their velocities. The 2nd row in Table 4 shows this clearly. Further highlighting how our approach leverages velocity data encoded in the RDIs efficiently.

Push and *pull* are performed in opposing directions. Again this directionality is captured explicitly in the RDIs. However, in our small scale experiment (row 3 of Table 4) *pull* was recognized poorly.

DISCUSSION & FUTURE WORK

We have introduced a novel method for the detection of *dynamic* gestures based on short-range radar in general and Google's Soli sensor in particular. Our model consisting of an end-to-end trained CNN and RNN combination achieves high accuracies on a large dataset and across different users. Furthermore, we have demonstrated that frame-to-frame approaches – somewhat of a standard in the HCI literature – face inherent challenges and that it is very important to model the dynamics of changing hand-pose configurations.

Our implementation runs in real-time on commodity PC hardware at 140 Hz. With the current successes of deep-learning methods in many application domains it is conceivable that special hardware[‡] will become available soon that would allow for interactive use even on mobile and wearable devices.

[‡]`http://www.nvidia.com/object/deep-learning.html`

In terms of future works, a natural follow-up research direction is to explore an end-to-end framework which resolves gesture spotting and classification jointly as demonstrated in [25] which is based on accelerometers or gyroscopes. We also want to highlight two interesting aspects of the sensor that we have not yet explored. First, while we have interpreted the sensor as gesture sensor, it is also possible to use millimeter wave radar for traditional ranging applications (tracking targets such as fingertips continuously in 3D). Another interesting aspect is that of RF waves penetrating occluding materials. This may be leveraged to sense through objects such as items of clothing or other stationary objects in the line-of-sight to the sensor. While we have not explored this in-depth we would like to highlight the similarities to recent work exploiting FMCW RF technology to coarsely 'image' users through a wall [1]. Another interesting research direction would be material recognition as shown in [17].

CONCLUSION

In this paper we have proposed the first gesture recognition technique capable of detecting a rich set of *dynamic* gestures based on high-frequency, short-range radar. Our technique is based on an end-to-end trained combination of deep convolutional and recurrent neural networks. The algorithm achieves high recognition rates (avg 87%) on a challenging gesture set including 11 gestures and across 10 users. To foster future work based on Google's Soli sensor and other related platforms we will release the pre-trained network architecture and training dataset publicly.

ACKNOWLEDGMENTS

We are grateful to Patrick Amihood, Erik Olson and Nick Gillian for engineering support, to Cécile Edwards-Rietmann for narrating the video. We also thank all the participants for their time and efforts in collecting gesture data.

REFERENCES

1. Adib, F., C.-Y. Hsu, H. Mao, D. Katabi, and F. Durand. Capturing the human figure through a wall. *ACM Transactions on Graphics (TOG)* (2015).
2. Butler, A., S. Izadi, and S. Hodges. SideSight: multi-touch interaction around small devices. In *Proc. ACM UIST*, 2008, 201–204.
3. Chen, K.-Y., K. Lyons, S. White, and S. Patel. uTrack: 3D input using two magnetic sensors. In *Proc. ACM UIST*, Oct. 2013, 237–244.
4. Cohn, G., D. Morris, S. Patel, and D. Tan. Humantenna: Using the body as an antenna for real-time whole-body interaction. In *Proc. ACM CHI*, 2012.
5. Darrell, T., and A. Pentland. Space-time gestures. In *Proceedings CVPR'93.*, IEEE, 1993.
6. Du, Y., W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *Proc. IEEE CVPR*, 2015.
7. Erol, A., G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly. Vision-based hand pose estimation: A

review. *Comput. Vision Image Understanding 108*, 1-2 (Oct. 2007), 52–73.

8. Goc, M. L., S. Taylor, S. Izadi, C. Keskin, M. Le Goc, S. Taylor, S. Izadi, and C. Keskin. A Low-cost Transparent Electric Field Sensor for 3D Interaction on Mobile Devices. In *Proc. ACM CHI*, Apr. 2014, 3167–3170.
9. Graves, A. *Supervised sequence labelling*. Springer, 2012.
10. Graves, A., A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *IEEE ICASSP*, IEEE, 2013.
11. Gupta, S., D. Morris, S. Patel, and D. S. Tan. SoundWave: using the doppler effect to sense gestures. In *Proc. ACM CHI*, 2012.
12. Hansen, C. J. Wigig: Multi-gigabit wireless communications in the 60 ghz band. *Wireless Communications, IEEE* (2011).
13. Harrison, C., D. Tan, and D. Morris. Skinput: Appropriating the Body As an Input Surface. In *Proc. ACM CHI*, 2010.
14. He, K., X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385* (2015).
15. Hinton, G. et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE* (2012).
16. Hong, P., M. Turk, and T. S. Huang. Gesture modeling and recognition using finite state machines. In *Automatic face and gesture recognition, 2000. proceedings. fourth ieee international conference on*, IEEE, 2000.
17. Hui-Shyong, Y., F. Gergely, S. Patrick, H.-B. David, and Q. Aaron. Radarcat: Radar categorization for input & interaction. In *Proc. ACM UIST*, 2016.
18. Hwang, S., M. Ahn, and K.-y. Wohn. MagGetz: customizable passive tangible controllers on and around conventional mobile devices. In *Proc. ACM UIST*, 2013.
19. Keskin, C., F. Kiraç, Y. E. Kara, L. Akarun, F. Kirac, Y. E. Kara, L. Akarun, and F. Kiraç. Hand Pose Estimation and Hand Shape Classification Using Multi-layered Randomized Decision Forests. ECCV'12, Springer, 2012, 852–863.
20. Kim, D., O. Hilliges, S. Izadi, A. D. Butler, J. Chen, I. Oikonomidis, and P. Olivier. Digits: Freehand 3D interactions anywhere using a wrist-worn gloveless sensor. In *Proc. ACM UIST*, 2012, 167.
21. Kratz, S., and M. Rohs. HoverFlow: expanding the design space of around-device interaction. In *MobileHCI*, 2009.
22. Lee, H.-K., and J. H. Kim. An HMM-based threshold model approach for gesture recognition. *IEEE PAMI* (1999).
23. Lien, J., N. Gillian, P. Amihood, and I. Poupyrev. Soli: Ubiquitous Gesture Sensing with Millimeter Wave Radar. *ACM Transactions on Graphics* (2016).
24. Neverova, N., C. Wolf, G. W. Taylor, and F. Nebout. Multi-scale deep learning for gesture detection and localization. In *Workshop at the European Conference on Computer Vision*, Springer, 2014.
25. Ordóñez, F. J., and D. Roggen. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors* (2016).
26. Pu, Q., S. Gupta, S. Gollakota, and S. Patel. Whole-home Gesture Recognition Using Wireless Signals. In *Proc. MobiCom*, 2013.
27. Pu, Q., S. Gupta, S. Gollakota, and S. Patel. Gesture Recognition Using Wireless Signals. *GetMobile: Mobile Comp. and Comm.* (2015).
28. Rekimoto, J. GestureWrist and GesturePad: unobtrusive wearable interaction devices. In *IEEE ISWC*, 2001.
29. Saponas, T. S., D. S. Tan, D. Morris, R. Balakrishnan, J. Turner, and J. A. Landay. Enabling always-available input with muscle-computer interfaces. In *Proc. ACM UIST*, no. 38, 2009.
30. Sato, M., I. Poupyrev, and C. Harrison. Touché: Enhancing touch interaction on humans, screens, liquids, and everyday objects. In *Proc. ACM CHI*, 2012.
31. Sharp, T. et al. Accurate, Robust, and Flexible Real-time Hand Tracking. In *Proc. ACM CHI*, 2015.
32. Shen, X., G. Hua, L. Williams, and Y. Wu. Dynamic hand gesture recognition: An exemplar-based approach from motion divergence fields. *Image and Vision Computing* (2012).
33. Simonyan, K., and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems*, 2014.
34. Song, J., G. Sörös, F. Pece, S. R. Fanello, S. Izadi, C. Keskin, and O. Hilliges. In-air Gestures Around Unmodified Mobile Devices. In *Proc. ACM UIST*, 2014.
35. Taylor, S., C. Keskin, O. Hilliges, S. Izadi, and J. Helmes. Type-Hover-Swipe in 96 Bytes: A Motion Sensing Mechanical Keyboard. In *Proc. ACM CHI*, 2014.
36. Werbos, P. J. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE* (1990).
37. Wilson, A., and S. Shafer. XWand: UI for intelligent spaces. In *Proc. ACM CHI*, 2003.
38. Zhai, S., P. Milgram, and W. Buxton. The influence of muscle groups on performance of multiple degree-of-freedom input. In *Proc. ACM CHI*, 1996.
39. Zhao, C., K.-Y. Chen, M. T. I. Aumi, S. Patel, and M. S. Reynolds. 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*, ACM, 2014.