Sensors and Materials, Vol. 32, No. 9 (2020) 2981–2998
MYU Tokyo

S & M 2317

Acoustic-sensing-based Gesture Recognition Using Hierarchical Classifier

Miki Kawato and Kaori Fujinami*

Department of Computer and Information Sciences, Tokyo University of Agriculture and Technology,
2-24-16 Naka-cho, Koganei, Tokyo 184-8588, Japan

(Received March 23, 2020; accepted May 25, 2020)

Keywords: gesture recognition, acoustic sensing, machine learning, hierarchical classifier, feature engineering

A gestural input to control artifacts and access the digital world is an essential part of highly usable systems. In this article, we propose a gesture recognition method that leverages the sound generated by the friction between a surface such as a table and a finger or pen, in which 17 different gestures are defined. The gesture recognition process is regarded as a 17-class classification problem; 89 classification features are defined to represent the envelope of each input sound, while a hierarchical classifier structure is employed to increase the accuracy of confusable classes. Offline experiments show that the highest accuracy is 0.954 under a condition where the classifiers are customized for each user, while an accuracy of 0.854 is obtained under a condition where the classifiers are trained without using the data from test users. We also confirm the effectiveness of the hierarchical classifier approach compared with a single-flat-classifier approach and that of a feature engineering approach compared with a feature learning approach. The information of individual features is also presented.

1. Introduction

Getting an input from a user is essential to control artifacts and access the digital world, whether explicitly or implicitly. A remote controller has been conventionally used to realize remote input into electronic devices such as home appliances, while a keyboard, computer mouse, and touch input have been popular for graphical user interfaces (GUIs); however, the demand for instant access to home appliances and digital services, as well as to highly usable systems, is increasing. Additionally, advances in multimedia processing technologies such as speech recognition[1,2] and image recognition,[3,4] as well as sensor and embedded system technologies such as wearable devices[5–7] and smart objects,[8,9] allow users to talk to or gesture with devices, and thus to give instructions (input) to a computer environment in the same way as in human-to-human communication. In this article, we propose an acoustic-sensing-based gesture recognition method that allows input in a natural and ubiquitous manner for use as the remote controller of a media player.

*Corresponding author: e-mail: fujinami@cc.tuat.ac.jp
https://doi.org/10.18494/SAM.2020.2878

ISSN 0914-4935
https://myukk.org/

Google Home[1] and Amazon Echo[2] operate various home appliances in a room using a human voice as the input and have great potential to be used as general-purpose input methods that may replace keyboards and traditional remote controls; however, people may be reluctant to talk to a machine, and it may take some time before everyone can accept such machines.[10] Handwriting recognition on an arbitrary input surface of a smartphone or tablet is a familiar method that can be incorporated without discomfort; however, this method cannot be used unless the terminal is at hand, and the size of the input surface depends on the size of the terminal. Thus, attention has been focused on gesture recognition, which allows input by moving an arm or finger in free space. A wearable device approach requires a user to wear a device, which allows relatively free input; ring-like devices such as MagicRing[5] and TypeRing[6] and a device mounted on an upper arm such as MyoKey[7] are often used. In particular, a ring is a familiar accessory, although rings are not necessarily worn by everyone and are not considered suitable for daily use such as to control home appliances. Gestures can also be recognized using sensors installed in surrounding environments, such as cameras and depth sensors;[3,4] however, Oulasvirta *et al.* showed that camera-based methods can give users the feeling of being monitored, thereby infringing their privacy.[11] Thus, there is still an issue with wearable-sensor-based and environment-based gesture recognition for practical use in daily life.

By contrast, a smart object approach might be effective as a method of suppressing privacy intrusion and reducing barriers for input, in which sensors are attached on or embedded into objects used in daily life. Zhang *et al.*[8] demonstrated Electrick, a touch input on a dedicated surface on which a conductive material was placed. In a different study, a pressure-sensitive surface was realized by installing load cells under the four corners of a surface, i.e., a table or an entire floor, to detect changes in the pressure applied to the surface,[9] which showed the possibility of providing information and services through natural actions. Knocki[12] is a commercially available knock-based remote control that is attached on a wall, door, table, or another surface and detects pressure when it is knocked on. Desks and walls are considered to be suitable for smart object-based gestural input because they are ubiquitous objects in indoor living spaces. Therefore, we take this approach as a means of natural input. In particular, gestures on arbitrary planes are detected by analysis of the sound caused by the friction between the surface and a finger or pen.

The use of acoustic sensing as an input to digital environments has been reported in a number of papers. Acoustic sensing can be categorized into three approaches: special-surface-based, active-sensing-based, and passive-sensing-based. The special-surface-based approach analyzes the acoustic signal generated when the user rubs a specially designed surface.[13,14] This approach enables a stable acoustic signal to be generated; however, the input area is limited to the place where the surface is deployed. The active-sensing-based approach also generates a particular space where the environment can be controlled by the system. The low-latency acoustic phase (LLAP),[15] a device-free gesture-tracking scheme based on the phase change of the acoustic wave from an audio speaker, utilizes a microphone and an audio speaker in a commercial smartphone to measure the movement direction and distance of a hand in front of the terminal. The LLAP is also sensitive to the place where the gesture is performed. By

contrast, the passive-sensing-based approach generally has a more relaxed condition of the place of input. The above-mentioned Knocki also fits into this category. The input to Knocki is very simple, i.e., the user simply knocks on a surface, which may restrict the variety of control. Scratch input[16] uses the unique sound when a fingernail is dragged over a textured surface, in which a machine-learning technique is utilized to discriminate six gestures using peak count and amplitude variation as features: single and double taps, as well as swipes with single ("I"), double ("V"), triple ("N"), and quad ("W") swipes. By contrast, we aim to recognize 17 gestures by using carefully designed features and a hierarchical classifier structure.

The remainder of this article is organized as follows. Section 2 describes the gesture recognition pipeline with gesture data collection. The evaluation methodology is also described. Section 3 describes the result of evaluation including the classification accuracy, the difference between a hierarchical classifier approach and a flat (single) classifier approach, and the effectiveness of classification features. Finally, Sect. 4 concludes the article.

2. Materials and Methods

2.1 Gestures

We specified 17 gestures that can be used to control a media player as shown in Fig. 1. Here, the red dots in the figure indicate the starting point of each input and the single red dot in the bottom right indicates a knock.

2.2 Dataset

2.2.1 Data acquisition system

The basic idea of the gesture recognition system is to discriminate characters and figures on the basis of acoustic wave patterns. The data collection system employs two sensors: a microphone and a piezoelectric sensor. The microphone is used to acquire audio signals propagating in the air, while the piezoelectric sensor is used to identify each segment of the

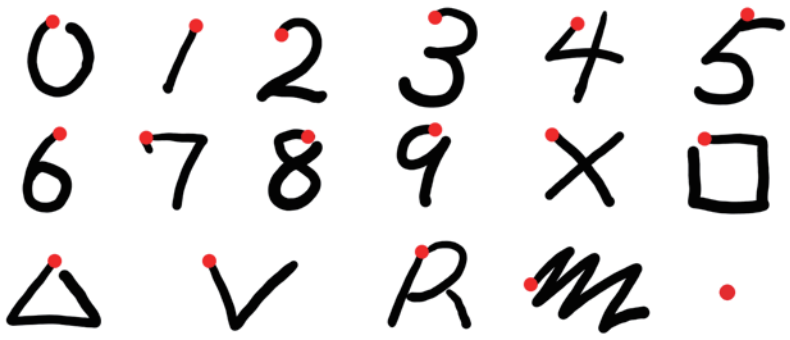


Fig. 1. (Color online) Supported gesture set.

gesture input based on the vibration signal propagated during the input, which is described in Sect. 2.3.1. The sensors are connected to a Raspberry Pi 3 (RP) microcontroller, which performs sampling at 44100 and 500 Hz for the microphone and piezoelectric sensor, respectively, with samples stored in an SD card in the RP. Note that the piezoelectric sensor is connected to the RP via an operational amplifier and an A/D converter.

The decision to use two sensors was basically to realize a low-cost acquisition system. It might also be possible to use a high-precision wall microphone to serve as a data source for both gesture input and gesture segmentation without using a microphone to acquire an air-propagated signal; finding the optimum sensor configuration is a future task. In this article, we focus on the recognition feature design and structure of a multiclass classifier, rather than the configuration of the sensing system.

2.2.2 Data collection

The gesture dataset was prepared in two phases. First, we assessed the effect of various conditions such as the surface material, the status of the input surface, and the distance to the sensors to reduce the possible combinations of conditions and thus the burden of the data provider, i.e., the human subject.[17] The surface materials we tested are common surfaces of furniture, i.e., (1) urethane paint, (2) printed veneer, and (3) melamine decorative board, which have different sensations of touch and thus may make different friction sounds when touched. Regarding the status of the input surface (a table), we tested under three conditions: (1) no objects on the surface apart from the acquisition device, (2) a low-density condition (a cup, a laptop, and a smartphone in addition to the acquisition device), and (3) a high-density condition (a wide variety of commonly used items in addition to the objects under the low-density condition). Furthermore, we changed the distance between the sensors and the input area, i.e., 10, 30, and 50 cm. One person performed all the input gestures, and the recognition accuracies among the gestures were compared. Here, we applied dynamic time warping (DTW) as a naïve gesture recognition method. To summarize our results, we confirmed large differences in the accuracy for different surface materials; however, we found that the status of the surface does not have any impact on the recognition accuracy. We also found no difference in recognition accuracy among the three different distances between the input area and the sensors.

The second phase was to collect data for training and testing of the proposed method. On the basis of the results of the above-mentioned preliminary experiment under various input conditions, we decided to use a melamine decorative board on which only the acquisition device was placed. Fifteen persons (eight males and seven females in their 20s) participated in the data collection. They performed the input gestures using a nib held in their dominant hand (14 persons were right-handed, and one person was left-handed). No directions were given regarding the speed of drawing. Each gesture was performed about 100 times, giving about 1500 trials per gesture. A photograph taken during data collection is shown in Fig. 2. Note that the microphone was covered with a polystyrene bowl in Fig. 2, which was to cut off ambient noise from far away.

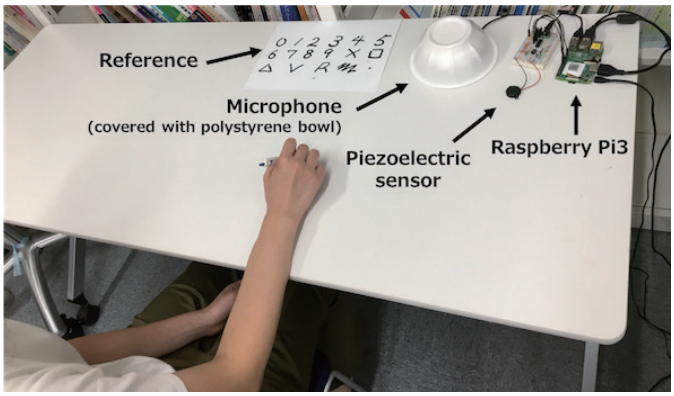


Fig. 2. (Color online) Photograph taken during data collection.

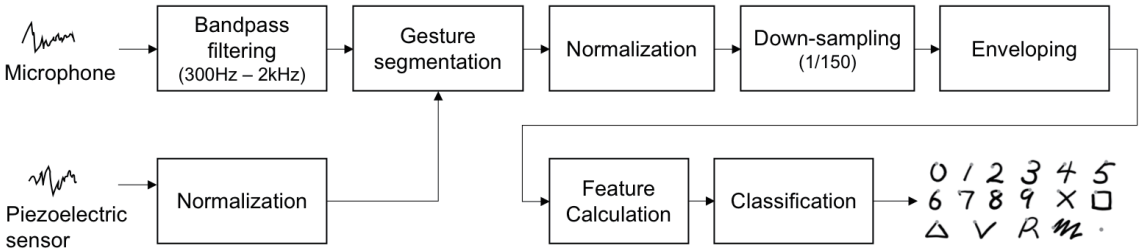


Fig. 3. Gesture recognition flow.

2.3 Gesture recognition method

In this section, the gesture recognition method is presented. The processing flow is presented in Fig. 3, which consists of four major components: preprocessing, gesture segmentation, feature calculation, and 17-class classification. In the following section, each component is described, with special focus on feature calculation.

2.3.1 Preprocessing

The preprocessing consists of filtering, gesture segmentation, normalization, down-sampling, and enveloping subprocesses. Environmental noise in raw data obtained from the microphone was removed by finite impulse response (FIR) bandpass filtering whose lower and upper cutoff frequencies were 300 and 2000 Hz, respectively. These frequencies were determined by a preliminary experiment.

The data stream contains periods of time that are not related to input gestures. Since gesture recognition is performed only for the period of time of input gestures, a segmentation process is required. To find the start and end points of one input gesture, we use the data from the piezoelectric sensor because significant vibration is observed during the input, especially at the

initial contact with the surface. Also, the piezoelectric sensor is not affected by background noise transmitted in the air. Thus, we consider it easier to identify the period of input than the signal from a microphone. We empirically set particular thresholds for the amplitude and for the interval of significant impact on the surface to judge if a particular period of time represents an input gesture.

Since the audio data are sampled at 44100 Hz, the amount of data for feature calculation is reduced by down-sampling, in which the root mean square (RMS) value is calculated per window with 150 samples. This means that the length of a segment is compressed to 1/150. Then, an envelope of the down-sampled segment is calculated as a target of feature calculation using the Hilbert transform. Note that normalization processes are carried out for the microphone signal and piezoelectric sensor signal to fit the range of values between 0.0 and 1.0.

2.3.2 Feature calculation

A gesture is recognized by comparing it with an input feature vector calculated from the segmented audio data. Designing powerful features is an important task in realizing a recognition system with high accuracy. In total, we defined 89 features, which are summarized in Table 1.

Statistical values such as duration, mean, standard deviation, and variance are used. In addition to calculating these features in an entire segment, we split the segment into two parts at the second local minimum in the segment, and the statistical features are calculated from the first half (FH) and second half (SH) of the segment and the entire (ALL) segment based on the findings in Ref. 18. The second local minimum was chosen as the split point due to

Table 1
Definitions of features.

Name	Number	Type	Description
$dur_{\{FH\|SH\|ALL\}}$	3	STAT	Duration of a particular time period (FH, SH, and ALL)
$max_{\{FH\|SH\|ALL\}}$	3	STAT	Maximum value in a particular time period (FH, SH, and ALL)
$mean_{\{FH\|SH\|ALL\}}$	3	STAT	Mean value in a particular time period (FH, SH, and ALL)
$sdev_{\{FH\|SH\|ALL\}}$	3	STAT	Standard deviation in a particular time period (FH, SH, and ALL)
$var_{\{FH\|SH\|ALL\}}$	3	STAT	Variance in a particular time period (FH, SH, and ALL)
n_max	1	MAX	Number of local maximums in the segment
v_max_i	5	MAX	Value of ith local maximum from the beginning of the segment
t_max_i	5	MAX	Normalized elapsed time at ith local maximum from the beginning of the segment
$d\_max_{i,i+1}$	4	MAX	Time difference between t_max_i and t_max_{i+1}
v_min_i	6	MIN	Value of ith local minimum from the beginning of the segment
t_min_i	6	MIN	Normalized elapsed time at ith local minimum from the beginning of the segment
$d\_min_{i,i+1}$	5	MIN	Time difference between t_min_i and t_min_{i+1}
t_mid_i	10	MID	Normalized elapsed time at ith midpoint from the beginning of the segment
d_mid_i	5	MID	Time difference between t_mid_k and t_mid_{k+1} that contains at least one maximum
$fluc_i$	10	FLUC	RMS of gradients in ith sub-segment
d_dtw_c	17	DTW	DTW distance between an input signal and a representative one for class c

FH and SH: first half and second half of a segment, respectively, ALL: entire segment
STATS: statistics-based, MAX: local-maximum-based, MIN: local-minimum-based, MID: midpoint-based
FLUC: fluctuation-based, DTW: DTW-based features

our observation that FH and SH tend to represent the first and second strokes of two-stroke gestures, respectively, such as "4" and "X".

Some of the features are specified so that the acoustic waveform of each gesture can be represented by listing prominent points such as local extremums (local maximums and local minimums), their corresponding times (normalized elapsed times from the beginning of the segment), and the time differences between successive local maximums and local minimums. A midpoint, which is defined as the value of a sample midway between a local minimum and the next local maximum, is defined, and normalized elapsed times that give midpoints are used. The time difference between two successive midpoints that contain at least one maximum is introduced to represent the sharpness of the peak. Figure 4 illustrates the notion of these features designed to represent the shape of the gesture waveform. Note that the maximum numbers of local maximums, local minimums, and midpoints are assumed to be five, six, and ten, respectively, on the basis of the observation of the collected data. Shortages are padded with zeros.

We also define a feature called fluctuation (FLUC) by the RMS of gradients. A slow change in the acoustic energy gives a small value of FLUC, which takes a value of zero in the case of motion with a constant velocity. Thus, we consider that FLUC can represent the complexity of an input gesture. An entire segment is divided into 10 subsegments, and FLUC is calculated for each subsegment.

Furthermore, the DTW distances between an input signal and representative signals of each gesture are introduced, in which three representative signals are randomly chosen for each gesture class in the entire dataset. The effectiveness of each feature is evaluated in Sect. 3.4.

2.3.3 Classification

The relationship between the input feature vectors and the input gesture classes is learned by supervised machine learning in the training phase. Then, the label of an input gesture is predicted from candidate labels of the 17 gestures by taking an unlabeled feature vector as an input to the classifier in an operation phase.

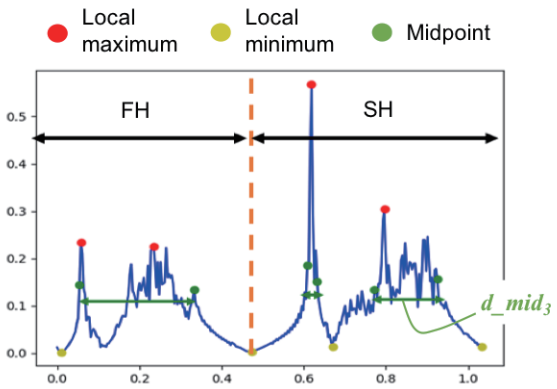


Fig. 4. (Color online) Definitions of features representing gesture signal waveforms.

We tested various classifier models, as well as two types of classifier structure: a single classifier (flat classifier) and a hierarchical classifier (see Fig. 5). Hierarchical classifier approaches are often used to realize extensible classification systems[19] and to filter out unnecessary activities.[20,21] In this work, the reason for using a hierarchical classification is to improve the classification performance. The hierarchy is determined empirically from the classification result of a single classifier. Figure 6 shows a confusion matrix when the RandomForest (RF) classifier was used. From the figure, we confirm that confusion exists in some gesture classes, which include classes between "0" and "6" and between "7" and "R". Thus, we propose the hierarchical classifier structure shown in Fig. 5. The top layer consists of nine classes, four of which merge two or six individual classes on the basis of the ease of confusion. By contrast, the bottom-layer classifiers are responsible for the detailed classification of merged classes.

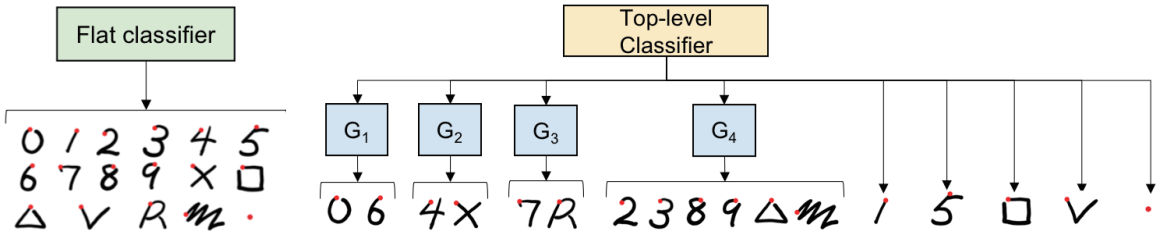


Fig. 5. (Color online) (a) Flat classifier and (b) hierarchical classifier structures in experiment.

Predicted label

Answer label	0	1	2	3	4	5	6	7	8	9	V	R	X	△	□	～	·
0	355	110	125	123	1	0	469	5	90	53	143	0	0	20	0	11	0
1	46	1413	19	0	0	2	10	0	2	1	13	0	0	0	0	0	10
2	63	57	762	19	2	32	29	1	118	112	122	0	0	159	0	39	0
3	59	1	57	752	45	3	123	2	77	58	158	0	42	43	1	98	0
4	0	2	0	6	1209	55	0	0	2	4	16	1	210	8	5	1	0
5	0	0	19	2	53	1305	0	33	11	17	7	10	9	29	16	6	0
6	365	114	64	99	0	1	564	2	68	128	75	0	2	26	0	0	0
7	3	8	1	2	1	4	0	1383	3	0	8	65	5	28	4	0	0
8	78	1	148	86	1	2	101	6	750	47	14	4	2	116	0	163	0
9	20	3	165	53	21	20	26	1	55	906	62	0	7	129	2	49	2
V	85	28	114	155	4	5	32	15	9	26	1031	0	9	2	0	0	0
R	1	0	2	2	1	2	0	97	16	0	0	1279	9	0	91	18	0
X	0	2	0	5	90	8	1	3	0	1	18	1	1386	0	1	0	0
△	8	4	152	22	3	6	28	91	80	204	43	5	2	850	1	14	0
□	0	1	1	0	1	12	0	21	0	0	0	108	9	0	1348	14	0
～	6	1	24	74	1	2	10	12	97	60	7	19	1	25	9	1167	0
·	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1509

Fig. 6. (Color online) Confusion matrix obtained by flat classifier structure using RF to consider grouping in proposed hierarchical classifier structure.

2.4 Experimental methodology

The objectives of the experiments are (1) to validate the effectiveness of the hierarchical classifier approach, (2) to clarify the difference in the classifier models, (3) to evaluate the effectiveness of the feature engineering approach for a limited data size, and (4) to evaluate the effectiveness of the features. To build an offline experimental environment, we utilized the Python programming language (ver. 3.6) with the SciPy scientific computing library (ver. 1.3), scikit-learn machine learning toolkit (ver. 0.19), and Keras neural network library (ver. 2.3). The envelope of the audio signal is obtained using `scipy.signal.hilbert`. The local minimum and local maximum are calculated using `scipy.signal.argrelmin` and `scipy.signal.argrelmax` with a parameter of `order` = 40, respectively. Here, `order` represents the number of points used to find the extremums. The larger the value of `order`, the more macroscopic the search range of the local extremum becomes. We empirically set the value to 40.

2.4.1 Classifier models

Two types of traditional classifier model were utilized: the support vector machine (SVM) and RF methods, which are popular in the activity recognition community owing to their high classification performance. We utilized the default hyperparameters for these models because our aim is to compare the effectiveness of the hierarchical approach and the flat structure rather than find the best classifier. As the main parameter in SVM, a radial basis function (RBF) kernel was used, whereas in RF, the number of estimators, also known as the number of trees, was 100.

In addition, we tested a convolutional neural network (CNN) as a modern end-to-end learning approach that does not need to craft a feature set via feature engineering. Therefore, in the case of CNN, the enveloping and feature calculation processes (Fig. 3) are skipped. The structure of the CNN classifier is shown in Fig. 7, which consists of three convolution layers, two of which are followed by max-pooling layers and one dropout layer. The dimension of an input is 929, which is the maximum length of a down-sampled segment in an entire dataset,

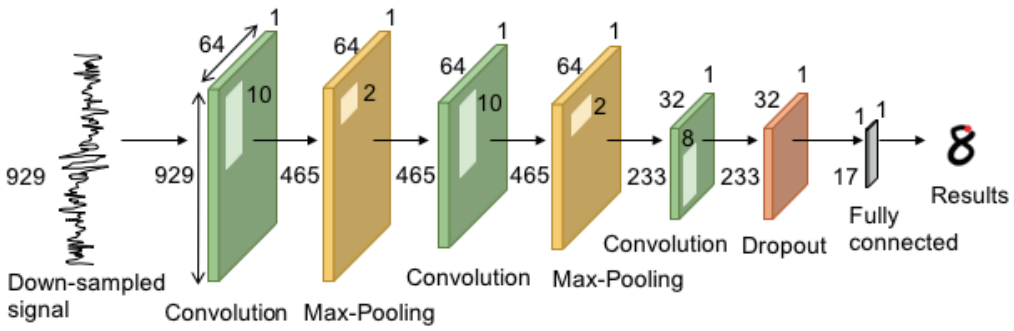


Fig. 7. (Color online) CNN structure.

and shorter segments are padded with zeros. The kernel size is ten or eight with a stride of one, while the number of filters is 64 or 32 and the pool size for max-pooling is set to two as shown in Fig. 7. The activation function in each convolution layer is the rectifier linear unit (ReLU). We do not intend to optimize the neural network structure; rather, we aim to investigate how well the feature engineering competes with the feature learning approach and how effective the hierarchical classification is.

2.4.2 Evaluation methods

Three evaluation methods are applied: macro 10-fold cross-validation (Macro-CV), micro 10-fold cross-validation (Micro-CV), and leave-one-subject-out cross-validation (LOSO-CV). Macro-CV is a cross-validation approach performed against a merged dataset from all 15 subjects. In this case, 9/10 of a dataset and 1/10 of a dataset are used for training and testing the classifier, respectively; the training dataset may contain 9/10 of the data from each person in theory, and hence the classifier "knows" about the subjects in the test data to some extent in advance. Macro-CV is considered to represent the average classification performance of a particular classifier model. By contrast, the result of Micro-CV is calculated by averaging the results of cross-validation against the dataset of each subject, which means that the classifier is perfectly customized for each test subject. Thus, the result indicates an upper limit of the classification performance. LOSO-CV is carried out by testing a dataset from a particular person with a classifier that is trained without a dataset from the person. Thus, LOSO-CV is regarded as the fairest and most practical test method among the three methods. In the experiment, these evaluation methods are used to confirm the average, upper limit, and lower limit of the classifiers. We use accuracy as the main performance metric, which is defined as the ratio of the number of correctly classified instances to the total number of instances. In addition, a confusion matrix is utilized to analyze the misclassification in detail.

The effectiveness of the feature engineering approach compared with an end-to-end learning approach, also known as the feature learning approach, is evaluated by changing the number of data used to train the RF and CNN classifiers by Macro-CV. A classifier is better if it achieves the same level of accuracy with less training data.

The importance of individual features is evaluated on the basis of information gain (IG). IG is commonly used in feature selection, in which the gain of information provided by a particular feature is calculated by subtracting the conditional entropy with that feature from the entropy under a random guess.[22] This means that a more informative feature has a higher IG.

3. Results and Discussion

3.1 Basic classification performance

The results of the three types of cross-validation are shown in Fig. 8, from which we can confirm the following points:

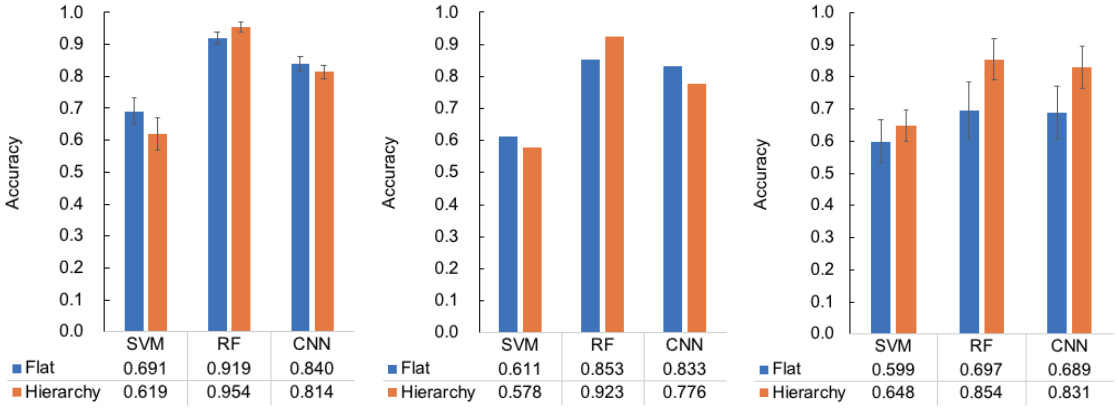


Fig. 8. (Color online) Results of (a) Micro-CV, (b) Macro-CV, and (c) LOSO-CV. The error bars in Micro-CV and LOSO-CV indicate the standard deviation among individual subjects.

- The results in Micro-CV showed the highest accuracy except for the case with SVM and CNN in LOSO-CV, where the highest accuracy (0.954) was realized by the hierarchical structure with RF in Micro-CV.
- The hierarchical classifier structure was found to be effective except for the case with SVM and CNN in Micro-CV and Macro-CV.
- RF showed the highest accuracy under all conditions.

The highest accuracy (0.954) in the Micro-CV case suggests that the gesture recognition system works reasonably well if the classification model is perfectly customized, although this would require the user's active involvement in the data collection process. One prospective solution is to apply an active learning technique[23] that allows the user to customize the classification system starting with a "semi-finished" classifier. As proposed by Fujinami *et al.*, selecting the classifier most compatible with the target user on the basis of a small number of samples would accelerate the learning process.[24] By contrast, the results in LOSO-CV show a practical performance where the classification model is independent of the test subject; the fact that all the accuracies with the hierarchical classifier structure in LOSO-CV are higher than those with the flat structure implies that the hierarchical approach is applicable to other classifier models such as the naïve Bayes and multilayer perceptron.

RF showed the highest accuracy among all evaluation methods, i.e., Micro-CV, Macro-CV, and LOSO-CV, and for both classifier structures, i.e., flat and hierarchical. It may be possible to improve the accuracy of the SVM hyperparameter tuning; however, RF has still an advantage because it has fewer hyperparameters than SVM; the parameter with the greatest contribution in RF is the number of trees (or estimators), and the accuracy is increased as the number increases. The accuracies in CNN were lower than those in RF, which we consider to be mainly due to the size of the training dataset, the effect of which is analyzed in Sect. 3.3.

3.2 Insight into hierarchical classification results

In this section, our analysis of the hierarchical classification is presented, in which we focus on the results in LOSO-CV as those of a practical condition. Figure 9 shows the accuracies of the top-level classifier, the second-level classifiers (G_1 to G_4), and the final result. Here, the top-level classifier deals with the classification of individual gesture classes such as "1" and "5", in addition to G_1 to G_4 as shown in Fig. 5(b), which sums nine classes. The accuracies of the top-layer classifiers obtained by SVM, RF, and CNN are 0.790, 0.854, and 0.863, respectively. As shown in the confusion matrix for the top-layer classifier obtained by RF in Fig. 10, a number of misclassifications in G_1 and G_4, as well as in "X" and G_4, are found. The top-level classifier plays a crucial role in the overall classification. Therefore, the top-level classifier must be improved by finding more effective features, tuning the hyperparameters, and/or reconsidering the grouping strategy for the second-layer classifiers by considering the overall classification accuracy.

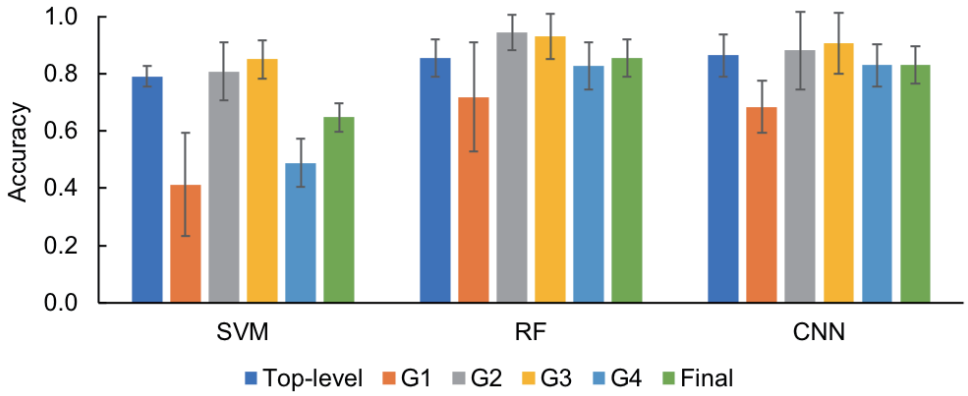


Fig. 9. (Color online) Accuracies of individual classifiers and final result in LOSO-CV.

Predicted label

Answer label	G_1	G_2	G_3	G_4	1	5	X	□	·
G_1	1640	1	3	1045	197	1	126	0	0
G_2	0	2903	8	53	4	46	13	8	0
G_3	5	12	2814	103	7	4	7	81	0
G_4	354	66	51	8339	62	14	210	6	0
1	90	0	0	18	1390	0	10	0	8
5	0	88	31	142	0	1240	5	11	0
X	130	16	14	503	23	1	828	0	0
□	0	9	134	20	1	10	0	1341	0
·	0	0	0	0	1	0	0	0	1509

Fig. 10. (Color online) Confusion matrix for top-layer classifier using RF.

Classifier G_1, which deals with "0" and "6", has the lowest accuracy among the four second-layer classifiers. These two characters look similar and inherently tend to be misclassified in a flat classifier structure, as shown in Fig. 6, in which "0" and "6" are misclassified into "6" and "0" 469 and 365 times, respectively. Additionally, the other gestures that are misclassified by the top-level classifier might be included in the classification of G_1 as described above.

However, the hierarchical classifier mostly worked well; Fig. 11 shows the confusion matrix for the overall result in LOSO-CV using RF (the detail of the green bar in the case of RF in Fig. 9). Comparison of the corresponding cells in the matrices of Figs. 6 and 11 shows that the accuracies for the gesture classes classified by the dedicated second-level classifiers, i.e., "0", "6", "4", "X", "7", "R", "2", "3", "8", "9", "△", and "~", were improved by 49.1% on average (minimum 4.3% and maximum 118.0%). By contrast, the accuracies of the classes directly classified by the top-level classifier were decreased by 5.4% on average, ranging from 0.0 to 19.7%. This strengthens the importance of the top-level classifier as a classifier for individual gesture classes, in addition to the grouped gesture classes.

Note that the classifiers took the same set of features and the hyperparameters were not tuned because we intended to compare the effectiveness of hierarchical gesture classification with that of a flat single classifier. Feature selection and hyperparameter tuning for each classifier might improve the accuracy. Furthermore, the structure of the hierarchical classifier, i.e., the grouping of gesture classes in each classifier, was empirically selected on the basis of the similarity of strokes when performing the gestures and the difficulty of discriminating with a flat classifier. Optimization of the classifier structure will be part of our future research.

Predicted label

Answer label

	0	1	2	3	4	5	6	7	8	9	V	R	X	△	□	~	·
0	774	95	170	139	1	0	0	2	49	87	88	0	0	81	0	19	0
1	43	1390	11	2	0	0	47	0	1	4	10	0	0	0	0	0	8
2	32	54	1292	0	2	7	40	0	0	0	88	0	0	0	0	0	0
3	31	1	0	1305	15	0	80	0	0	0	70	0	17	0	0	0	0
4	0	2	4	8	1429	42	0	1	2	13	1	0	0	8	6	3	0
5	0	0	12	23	74	1240	0	24	11	54	5	7	14	8	11	34	0
6	0	102	65	187	0	1	866	1	48	138	38	0	0	43	0	19	0
7	2	7	1	7	1	3	2	1442	8	2	7	0	3	25	3	2	0
8	87	1	0	0	0	2	45	0	1379	0	4	1	0	0	0	0	0
9	5	1	0	0	3	5	13	0	0	1443	28	1	22	0	0	0	0
V	67	23	154	168	0	1	63	13	53	88	828	1	16	36	0	4	0
R	1	0	0	2	1	1	0	0	11	15	0	1372	7	0	78	30	0
X	0	2	0	5	0	4	0	5	0	5	12	2	1474	4	2	1	0
△	13	4	0	0	2	0	6	40	0	0	20	0	5	1422	1	0	0
□	0	1	0	0	0	10	0	24	3	2	0	110	9	0	1341	15	0
~	1	1	0	0	0	0	1	2	0	0	0	7	0	0	5	1498	0
·	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1509

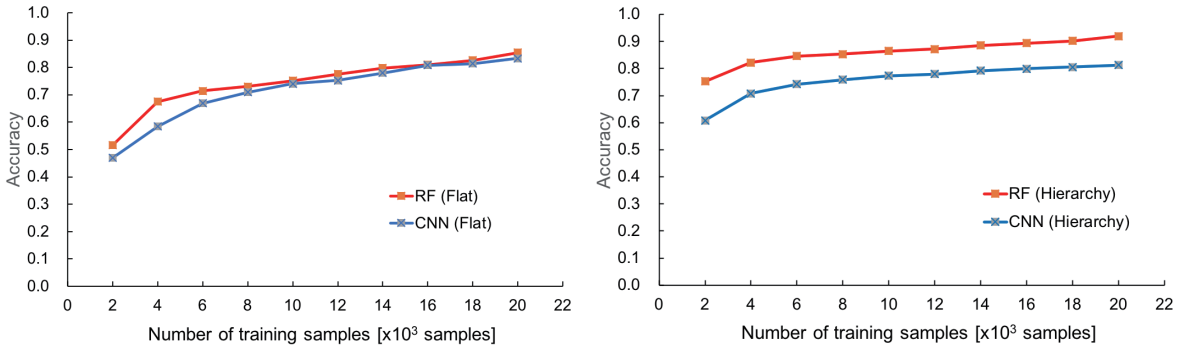
Fig. 11. (Color online) Confusion matrix with hierarchical classifier structure using RF.

3.3 Effectiveness of feature engineering approach under limited data size

Figure 12 shows the learning curves of RF and CNN classifiers obtained by Macro-CV evaluation, which correspond to the bars in Fig. 8(b). Figure 12(a) shows that the accuracy of the feature engineering approach is comparable to that of the feature learning approach, while the feature engineering approach has clearly better accuracy in the hierarchical classification as shown in Fig. 12(b). This implies that the proposed feature set is more effective with the hierarchical classifier structure than the features learned by CNN with the structure shown in Fig. 7 and the same number of training data. The same tendencies were expected to be observed for Micro-CV and LOSO-CV.

3.4 Importance of individual classification features

Figure 13 shows a boxplot that represents the informativeness of each feature, while Table 2 summarizes the IG of individual features.



(a) (b)

Fig. 12. (Color online) Learning curves of (a) flat classifier structures and (b) hierarchical classifier structures.

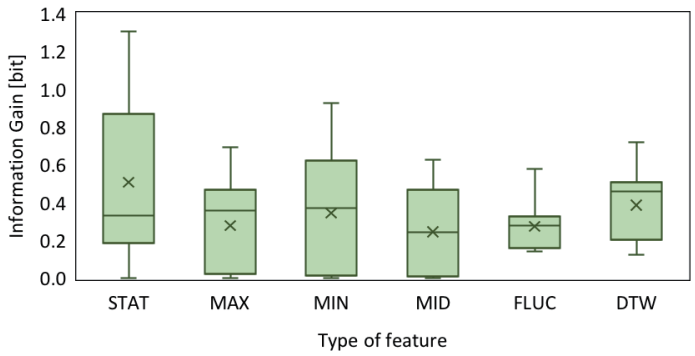


Fig. 13. (Color online) Boxplot of IG for different feature types.

Table 2
IG for each feature and order of addition to feature subset (CFS).

IG	Name	IG	Name	IG	Name	IG	Name
1.307	dur_{ALL}	0.468	d_mid_2	0.267	$sdev_{ALL}$	0.012	v_min_5
1.012	dur_{SH}	0.466	$d\_max_{2,3}$	0.267	var_{ALL}	0.011	t_mid_7
0.929	t_min_2	0.465	v_min_2	0.253	$fluc_6$	0.010	d_mid_4
0.871	$sdev_{SH}$	0.458	d_dtw_3	0.244	t_mid_1	0.008	t_mid_8
0.871	var_{SH}	0.447	v_max_2	0.233	d_dtw_{12}	0.000	d_mid_5
0.868	t_min_3	0.440	t_max_3	0.216	t_mid_6	0.000	$d\_max_{4,5}$
0.830	mean_{SH}	0.432	v_max_3	0.197	$mean_{FH}$	0.000	$d\_min_{5,6}$
0.812	v_min_3	0.429	t_min_4	0.187	max_{FH}	0.000	max_{ALL}
0.794	max_{SH}	0.400	d_mid_1	0.182	$fluc_2$	0.000	t_mid_{10}
0.782	$d\_min_{2,3}$	0.400	$d\_min_{3,4}$	0.174	d_dtw_{10}	0.000	t_mid_9
0.719	d_dtw_{17}	0.385	v_min_4	0.174	d_dtw_{14}	0.000	t_max_5
0.695	t_max_2	0.377	d_dtw_{13}	0.167	$fluc_4$	0.000	t_min_6
0.629	t_mid_2	0.372	t_min_1	0.157	$sdev_{FH}$	0.000	v_max_5
0.599	n_max	0.357	t_max_1	0.157	var_{FH}	0.000	v_min_6
0.589	t_mid_3	0.355	dur_{FH}	0.145	$fluc_{10}$		
0.579	d_dtw_2	0.332	$mean_{ALL}$	0.142	$fluc_5$		
0.577	$fluc_9$	0.329	$fluc_1$	0.135	d_dtw_8		
0.565	$d\_max_{1,2}$	0.327	$fluc_8$	0.123	d_dtw_4		
0.552	d_dtw_{16}	0.324	$fluc_7$	0.092	v_max_1		
0.530	t_mid_4	0.316	$d\_min_{1,2}$	0.049	v_min_1		
0.526	d_dtw_1	0.307	$fluc_3$	0.024	$d\_max_{3,4}$		
0.489	d_dtw_7	0.297	d_dtw_6	0.024	t_max_4		
0.488	d_dtw_5	0.293	t_mid_5	0.024	v_max_4		
0.483	d_dtw_{11}	0.286	d_dtw_{15}	0.012	$d\_min_{4,5}$		
0.482	d_dtw_9	0.286	d_mid_3	0.012	t_min_5		

The statistical features (STAT) are the most informative on average, although informativeness is diverse among the features, that is, IG ranges from 0.000 (max_{all}) to 1.307 (dur_{ALL}). Although the durations (dur_{ALL} and dur_{SH}) are not normalized and therefore may contain individual differences in the length of the stroke, they are the most and second-most informative features.

The features derived from SH are generally informative. This is probably because the first and second local minimums are observed just before and after the user makes contact with the surface, respectively, and because the majority of the gesture for a particular class is included in the region after the second minimum, that is, in SH.

The features that represent the shape of the envelope that have a prefix of "*t_*" are informative, especially the normalized elapsed time at the second and third local extremums. The reason why the third quartile and the minimum values of the feature types of MAX, MIN, and MID are low is that not all the input gestures have a large number of local maximums and minimums. That is, we set the upper limits of *t_max*, *t_min*, and *t_mid* to 5, 6, and 10, respectively, as shown in Table 1. Thus, these features may have a value of zero, i.e., the default value, regardless of the gesture class. The type FLUC is calculated against subsegments of 1/10 of the size of the entire segment. Therefore, we consider that such a "default value" problem did not occur and thus the deviation is small.

The median of the DTW-derived features is the highest and is higher than the average. This means that the majority of these features are very informative, while a small proportion of the features in the category are not so informative. As shown in Table 2, nine out of 17 DTW-derived features (*d_dtw*) appear in the top 30 informative features, and four of them are in the bottom 30. The DTW-derived features represent the proximity between the input gesture class and each class. Thus, it is straightforward to relate the value (DTW distance) with a particular class.

It should be noted that the collection of the k most informative features does not necessarily give high accuracy of the classifier. To find the best feature subset, the goodness of a particular feature subset should be evaluated in a systematic manner,[25] which may also contribute to improving the processing speed owing to the reduced dimension.

4. Conclusions

In this article, we proposed a gesture recognition method that leverages the sound caused by friction between a surface, e.g., table, and a finger or a pen, in which 17 gestures were supported. Two challenges exist: feature design and classifier structure. We primarily took a feature engineering approach and defined 89 features for gesture classification from the envelope of the acoustic signal so that they can represent the shapes of individual gestures. As the classifier structure, we employed a hierarchical structure with two layers, in which the first layer deals with nine types of gesture including groups of gestures that can be easily confused, e.g., "0" and "6", while the second layer focuses on discriminating these confusable gestures. The classification accuracy was evaluated using three models of classifiers, i.e., SVM, RF, and CNN. Offline experiments showed the following:

(1) The highest accuracy in a favorable situation (Micro-CV), where the gesture classifiers are trained by only the particular person's data, was 0.954, obtained by RF. By contrast, the highest accuracy in a practical situation (LOSO-CV), where the classifiers are trained by data excluding the user him/herself, was 0.854, obtained by RF.
(2) The hierarchical approach was generally more effective than the single (flat) classifier approach, especially under all evaluation conditions of RF.
(3) The feature engineering approach showed the greater effectiveness of the hierarchical classifier structure than the feature learning approach, i.e., CNN, for a training dataset of the same size.
(4) The analysis of the informativeness of individual features showed that the duration of a gesture segment was the most informative and that the normalized elapsed times of local minimums and local maximums are also informative. Additionally, the feature that represents the DTW distance had the highest median and a small variance.

It may be possible to improve the classification accuracy by selecting contributive feature subsets for each classifier from the current 89 features. Also, the high accuracy under the Micro-CV condition suggests that the accuracy can be greatly improved by asking the user to provide his or her own gesture data. Even if not all the data can be provided, an active-learning approach can incrementally improve the accuracy by selecting the most uncertain input as a

target of labeling by the user. The active-learning approach will be important for practical use because not only the user's habits but also the operating environment may affect gesture patterns. The hierarchical classifier structure showed better accuracy than the flat structure; however, a better hierarchy might exist. Automatic construction of the optimum hierarchy is a target of our future work.

References

1 Google: https://store.google.com/product/google_home (accessed March 2020).
2 Amazon: https://www.amazon.com/all-new-Echo/dp/B07R1CXKN7 (accessed March 2020).
3 K. C. Lim, S. H. Sin, C. W. Lee, W. K. Chin, J. Lin, K. Nguyen, Q. H. Nguyen., B. P. Nguyen, and M. Chua: Proc. 4th Int. Conf. Machine Learning and Soft Computing (ACM, 2020) 108. https://doi.org/10.1145/3380688.3380711
4 I. A. Stinghen, B. B. Gatto, J. L. S. Pio: Proc. XVII Brazilian Symp. Computer Games and Digital Entertainment (2018) 659.
5 L. Jing, Z. Cheng, Y. Zhou, J. Wang, and T. Huang: Proc. 12th Int. Conf. Mobile and Ubiquitous Multimedia (ACM, 2013) Article No. 30. https://doi.org/10.1145/2541831.2541875
6 S. Nirjon, J. Gummeson, D. Gelb, and K-H. Kim: Proc. 13th Annu. Int. Conf. Mobile Systems, Applications, and Services (ACM, 2015) 227. https://doi.org/10.1145/2742647.2742665
7 Y. Kwon, K. A. Shatilov, L-H. Lee, S. Kumyol, K. Y. Lam, Y-P. Yau, and P. Hui: Proc. IEEE Int. Conf. Pervasive Computing and Communications (IEEE, 2020).
8 Y. Zhang, G. Laput, and C. Harrison: Proc. 2017 CHI Conf. Human Factors in Computing Systems (ACM, 2017) 1. https://doi.org/10.1145/3025453.3025842
9 A. Schmidt, M. Strohbach, K. van Laerhoven, and H-W. Gellersen: Universal Access Theoretical Perspectives, Practice, and Experience. (Springer, Hidelberg, 2003) 263. https://doi.org/10.1007/3-540-36572-9_21
10 KDDI Corp.: https://news.kddi.com/kddi/corporate/newsrelease/2017/10/05/besshi2726.html#1-2 (accessed March 2020) (In Japanese).
11 A. Oulasvirta, A. Pihlajamaa, J. Perkiö, D. Ray, T. Vähäkangas, T. Hasu, N. Vainio, and P. Myllymäki: Proc. 2012 ACM Conf. Ubiquitous Computing (ACM, 2012) 41. https://doi.org/10.1145/2370216.2370224
12 Knocki: https://www.kickstarter.com/projects/knocki/knocki-make-any-surface-smart (accessed March 2020).
13 R. Murray-Smith, J. Williamson, S. Hughes, and T. Quaade: Proc. SIGCHI Conf. Human Factors in Computing Systems (ACM, 2008) 1299. https://doi.org/10.1145/1357054.1357257
14 J.-E. Kim, J. Sunwoo, Y.-K. Son, D.-W. Lee, and I.-Y. Cho: Proc. CHI '07 Extended Abstracts on Human Factors in Computing Systems (ACM, 2007) 2495. https://doi.org/10.1145/1240866.1241030
15 W. Wang, A. X. Liu, and K. Sun: Proc. 22nd Annu. Int. Conf. Mobile Computing and Networking (ACM, 2016) 82. https://doi.org/10.1145/2973750.2973764
16 C. Harrison and S. E. Hudson: Proc. 21st Annu. ACM Symp. User Interface Software and Technology (ACM, 2008) 205. https://doi.org/10.1145/1449715.1449747
17 M. Kawato and K. Fujinami: Tech. Rep. IEICE No. HPB_No20_03 (2019) (in Japanese).
18 T. Ishikawa and K. Fujinami: ISPRS Int. J. Geo-Inf. **2016** (2016) 5. https://www.mdpi.com/2220-9964/5/10/182
19 H. Leutheuser, D. Schuldhaus, and B. M. Eskofier: PLOS ONE (2013) 8. https://doi.org/10.1371/journal.pone.0075196
20 T. T. Vu, A. Sokan, H. Nakajo, K. Fujinami, J. Suutala, P. Siirtola, T. Alasalmi, A. Pitkanen, and J. Röning: Proc. 2011 IEEE 17th Int. Conf. Embedded and Real-Time Computing Systems and Applications (IEEE, 2011) 138. https://ieeexplore.ieee.org/document/6029875/
21 A. Reiss, D. Stricker, and G. Hendeby: Proc. 2013 7th Int. Conf. Pervasive Computing Technologies for Healthcare and Workshops (IEEE, 2013) 25. https://eudl.eu/doi/10.4108/icst.pervasivehealth.2013.251928
22 I. H. Witten, E. Frank, and M. A. Hall, Data Mining: Practical Machine Learning Tools and Techniques, Ed. San Francisco (Morgan Kaufmann Publishers, CA, 2011) 3rd ed.
23 B. Settles: University of Wisconsin-Madison Department of Computer Sciences Technical Report (2009). https://minds.wisconsin.edu/bitstream/handle/1793/60660/TR1648.pdf
24 K. Fujinami, T. T. Vu, and K. Sato: Proc. 17th Int. Conf. Pervasive Intelligence and Computing (IEEE, 2019) 885. https://doi.org/10.1109/DASC/PiCom/CBDCom/CyberSciTech.2019.00162
25 M. A. Hall: Ph.D. Thesis, University of Waikato (1998). https://www.cms.waikato.ac.nz/~ml/publications/1999/99MH-Thesis.pdf

About the Authors

Miki Kawato received his B.S. and M.S. degrees in computer science from Tokyo University of Agriculture and Technology, Japan, in 2018 and 2020, respectively. His research interests are in machine learning and gesture recognition.

Kaori Fujinami received his B.S. and M.S. degrees in electrical engineering and his Ph.D. degree in computer science from Waseda University, Japan, in 1993, 1995, and 2005, respectively. From 2005 to 2006, he was a visiting lecturer at Waseda University. From 2007 to 2017, he was an associate professor in the Department of Computer and Information Sciences at Tokyo University of Agriculture and Technology (TUAT). Since 2018, he has been a professor at TUAT. His research interests are in machine learning, activity recognition, human–computer interaction, and ubiquitous computing. (fujinami@cc.tuat.ac.jp)