Bento Packaging Activity Recognition with Convolutional LSTM using Autocorrelation Function and Majority Vote

Atsuhiro Fujii, Kazuki Yoshida, Kiichi Shirai, Kazuya Murao

Abstract This paper reports Bento Packaging Activity Recognition Challenge by team "RitsBen" held in the International Conference on Activity and Behavior Computing (ABC 2021). Our approach used an autocorrelation function in the preprocessing to isolate the data since the dataset was given with repetitive activity. We then use a model that implements convolutional layers, and LSTM. The final decision is made by majority vote using sigmoid predictions output from all body parts. The loss is calculated using BCEWithLogitsLoss for each body part. The evaluation results showed that average accuracy of 0.123 were achieved among subjects 1, 2, and 3 in leave-one-subject-out manner. However, we did not achieve high accuracy as the possibility that the extraction of repetitive actions was not correct.

1 Introduction

This paper reports the solution of our team "RitsBen" to Bento Packaging Activity Recognition Challenge held at the International Conference on Activity and Behavior Computing (ABC2021). The goal of Bento Packaging Activity Recognition

Atsuhiro Fujii,
Graduate School of Information Science and Engineering, Ritsumeikan University, 1-1-1 Nojihigashi, Kusatsu, Shiga 525-8577, Japan

Kazuki Yoshida,
Graduate School of Information Science and Engineering, Ritsumeikan University, 1-1-1 Nojihigashi, Kusatsu, Shiga 525-8577, Japan

Kiichi Shirai,
Graduate School of Information Science and Engineering, Ritsumeikan University, 1-1-1 Nojihigashi, Kusatsu, Shiga 525-8577, Japan

Kazuya Murao,
Graduate School of Information Science and Engineering, Ritsumeikan University, 1-1-1 Nojihigashi, Kusatsu, Shiga 525-8577, Japan e-mail: murao@cs.ritsumei.ac.jp

Challenge is to distinguish activities taking place during each segment based on the motion data collected with motion capture sensors while performing Bento-box packaging tasks. Activity recognition is the process of automatically inferring what a user is doing based on sensor observations. Since activity recognition can enrich our lives by understanding the characteristics of human activity, there has been a lot of research on human activity recognition (HAR).

Lara et al. [7] surveyed the state of the art in HAR based on wearable sensors. Khan et al. [6] proposed an accelerometer-based HAR method. Bayat et al. [3] proposed a recognition system in which a new digital low-pass filter is designed. Anguita et al. [1] conducted a comparative study on HAR using inertial sensors in a smartphone. Attal et al. [2] presents a review of different classification techniques in HAR based on wearable inertial sensors. In recent years, methods that use neural networks to improve the accuracy of activity recognition have also been actively researched. Yang et al. [10] propose a systematic feature learning method based on deep convolutional neural networks (CNN) for HAR problem. Chen et al. [4] proposed a deep learning approach to HAR based on single accelerometer. Tsokov et al. [9] proposed an evolutionary based approach for optimizing the architecture of one dimensional CNNs for HAR. Dang et al. [5] introduced a classification of HAR methodologies, and shows advantages and weaknesses for methods in each category.

In this paper, we construct a network with convolutional layers and LSTM layer to recognize the activities and detect repetition using the autocorrelation function. The outputs of the model are sigmoid values, the number of the outputs is as much as the number of sensors. The final output is a majority vote on those sigmoid values.

2 Challenge

In this challenge, each team competes in the recognition accuracy of activities related to Bento-box packaging activities based on motion capture data. For more details, please refer to the article [8].

The data collected from four subjects (all males) who attached one motion capture system with 29 markers released by Motion Analysis Company[1]. The subjects packed three types of food according to the five different bento-box packaging scenarios. The subjects performed in two patterns of outward and inward, and five times each scenario. In total, 4 subjects $\times$ 5 scenarios $\times$ 2 patterns $\times$ 5 trials = 200 of data were collected. Training data contains data from three subjects (subjects 1, 2, 3) out of the four subjects and test data contains the data from the fourth subject (subject 4). Originally, the training dataset should be contained 150 trials for three subjects. However, activity 6 of subject 1 did not contain the fifth trial and the activity 6 and 9 of subject 2 contained the sixth trial. Therefore, the total number of training data was 151.

[1] https://motionanalysis.com

Table 1 shows a list of 10 different activity names, movement direction patterns, and activity labels; there are 5 different activity types, and 2 movement direction patterns, for a total of 10 types (5×2 = 10). The segment is stored and divided by subject, activity label, and trials. For example, `[subject_1_activity_1_repeat_1]` contains the data of subject 1 having performed the first trial of activity whose label number is 1. Each file contains 89 types of data: 3-axis (X, Y, Z) data measured from each of 29 markers, subject number, and activity label. The measurement time for each file ranges from 50 to 70 s. Note that due to the complicated setup of the motion capture sensor, missing measurement data and incorrect activity labels may be included.

Table 1 A list of 10 different activity names, movement direction patterns, and activity labels.

Activity name	Movement direction pattern	Activity label
Normal	Inward	1
	Outward	2
Forgot to put ingredients	Inward	3
	Outward	4
Failed to put ingredients	Inward	5
	Outward	6
Turn over bento-box	Inward	7
	Outward	8
Fix/rearranging ingredients	Inward	9
	Outward	10

Table 2 shows the number of recognized classes (10 classes in this challenge), the number of segments for each subject, the number of trials for each activity, and the maximum, mean, and minimum length of the segments.

Submissions from the participants will be evaluated by the accuracy of activity classification. The accuracy is given by $accuracy = \frac{P \cap G}{P \cup G}$; the number of correct labels predicted (logical product of prediction set P and groundtruth set G) divided by the number of total true and predicted labels (logical sum of P and G).

3 Method

This section describes the preprocessing to obtain the features from the raw data, the structure of the model, the loss function and the optimizer, the process of obtaining the activity labels from the predictions obtained by the one-hot vector, and implementation. Note that our method does not use motion capture data.

Table 2 Statistics of the dataset.

Subject	Activity label	# of recognized classes	# of segments	# of trials	Length		
					max	mean	min
1	1	10	49	5	5701	5311	5516
	2			5	6325	5120	5860
	3			5	6448	5956	6214
	4			5	6665	6257	6446
	5			5	6467	5955	6310
	6			4	5771	5427	5525
	7			5	5725	5323	5567
	8			5	6653	6307	6395
	9			5	6828	6252	6530
	10			5	6962	6269	6525
2	1	10	52	5	7660	6626	7037
	2			5	6788	6493	6667
	3			5	6827	6390	6610
	4			5	6933	6295	6564
	5			5	6690	6470	6608
	6			6	8441	7660	7996
	7			5	7719	7392	7511
	8			5	7870	7386	7667
	9			6	7434	1663	6343
	10			5	7420	6651	6343
3	1	10	50	5	6337	5972	6149
	2			5	6326	5724	5915
	3			5	5995	5582	5801
	4			5	5930	5596	5768
	5			5	5953	5523	5778
	6			5	6190	5402	5769
	7			5	6017	5408	5737
	8			5	5718	5223	5459
	9			5	5860	5357	5620
	10			5	5714	5351	5579
4	Unknown	Unknown	48	Unknown	5947	5024	5481

3.1 Preprocessing

Figure 1 shows the flow of preprocessing. The details of each process are described below.

- **Reading data process** reads raw data for each body part from a given dataset. The details of this dataset are described in Section 2.
- **Converting process** converts to velocity data from raw data. First, we interpolate the missing values in the raw data. The time when the missing data starts is t, the time when the missing data ends is $t+n$, the i-th missing time from the start time is $t+i$ $(0 \leq t \leq t+i \leq t+n \leq T)$, the raw data at t is $R(t)$, and the interpolated data at t is $I(t)$. $I(t+i)$ is calculated by $I(t+i) = R(t) + \frac{R(t+n)-R(t)}{n} \times i$. In case of missing data at the beginning or end of a segment, the data that up to the time when the data was available at all markers was deleted and interpolated. The

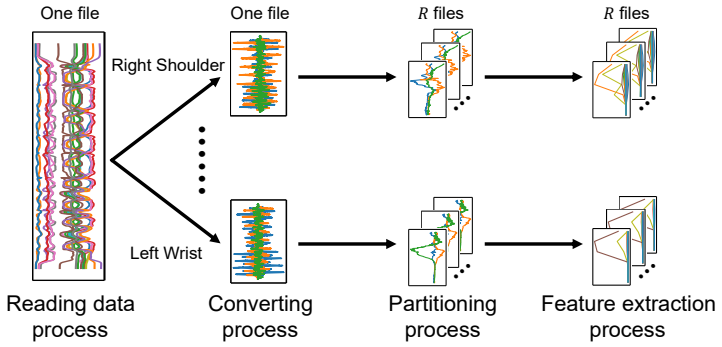

Fig. 1 Details of preprocessing. The first process is the raw data as it is provided. In the second process, the raw data is converted to velocity data. The partitioning process identifies and separates repetitive parts from the velocity data. The feature extraction process extracts 21-dimensional features from the separated velocity data.

interpolated data was used to calculate the velocity data. The interpolated data at time j $(1 \leq j \leq T)$ is $I(j)$, and the velocity data is $V(j)$, which is calculated by $V(j) = \frac{I(j)-I(j-1)}{0.01}$. Since the dataset frequency in this challenge was 100 Hz, the calculation was divided by 0.01.

- **Partitioning process** is the process of dividing the given data of approximately 60 s into data for one operation. The input is a single time-series velocity data, and the output is R time-series velocity data. R is the number of repetitive actions that were included. This time, we used the autocorrelation function. The autocorrelation function is the correlation between different points in a time series. When the time series data has periodicity, the autocorrelation function also shows a peak at the same period. Therefore, by applying the autocorrelation function to the data set which was obtained by repeating the same operation, and finding the local maximum value, we aimed to extract the data of one operation of period T. We summed up all the sensor values in the data set to form a single synthetic wave, and applied the autocorrelation function.
- **Feature extraction process** is to extract the characteristic data. Our method uses mean, variance, max, min, root mean square (RMS), interquartile range (IQR), and zero-crossing rate (ZCR) for the features. These features were extracted with a window size of 400 ms and an overlap of 50 ms. From these preprocessing, 7 features $\times$ 3 axes = 21 dimensions feature time series are obtained for one marker. We thought that upper body movements had a great influence on the activity. Therefore, we used only the data obtained from the markers attached to six parts of the upper body (Right Shoulder, Right Elbow, Right Wrist, Left Shoulder, Left Elbow, Left Wrist).

3.2 Model

The feature data created by the preprocessing is fed into our model. Figure 2 shows the structure of our model. The model consists of 1d convolutional layer, LSTM layer, Linear layer, and Sigmold layer. The models for six sensors are trained separately. The final activation layer determines one final prediction label from the six predictions. The details of each layer are described below.

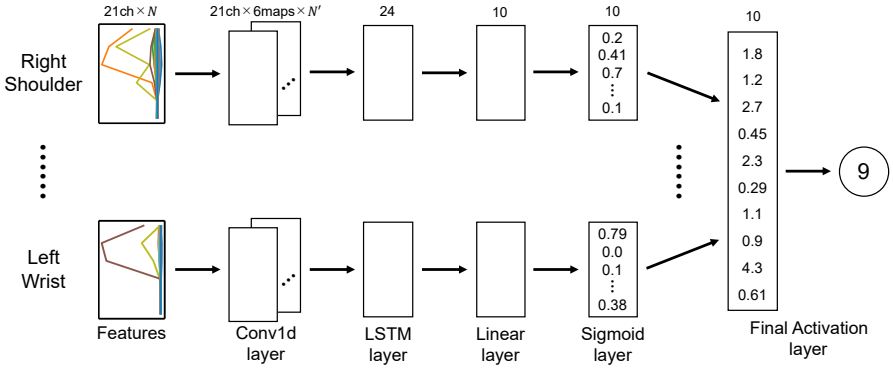

Fig. 2 Details of our model. The first layer is preprocessed 21-dimensional features. Conv1d layer is one dimensional convolutional layer. This layer accepts 21-dimensional time-series features and maps each dimension to six (i.e. 21-dimensional features × 6 maps = 126-channel). The LSTM layer consists of 24 hidden layers, accepts 126-channel time-series data, and outputs 24-dimensional tensors. The linear layer transforms 24-dimensional tensors into 10-dimensional ones. After applying the sigmoid function in the sigmoid layer, the six predictions are merged to obtain the final prediction result.

- **Conv1d layer** has an input of 21 channels × sequence length N and an output of 21 channels × map size M × sequence length N'. N is the length of hand crafted time-series feature data, which is shorter than the raw data. N' is the length of time-series data after the one dimensional convolution. This is equal to $N - K + 1$ (K is the kernel size). N and N' varies with the data because the dataset contains missing data. Kernel size K is set to 5. If N is 4, N' will be 0. Therefore, segments with N shorter than 5 are discarded and not fed into the model. Map size M is the number of filters and set to 6 × 21-dimensional features = 126. There are 6 filters for each channel, and the convolution is conducted for each channel.
- **LSTM layer** has an input of 126 channels × sequence length N' and an output of 24-dimensional tensors. This LSTM solves many to one task. We set the number of hidden layers to 24. The outputs obtained from the LSTM are not time-series data, but simply tensors.
- **Linear layer** has an input of 24-dimensional tensor and an output of 10-dimensional tensor which is the same as the number of activity classes.

- **Sigmoid layer** applies the sigmoid activation function to the 10-dimensional tensor. The output 10-dimensional tensor indicates the likelihood that the class is correct. This layer is not used in the training phase.

3.3 Loss Function and Optimizer

The model is trained by BCEWithLogitsLoss, which is a stable loss equipped with a Sigmoid function. Sigmoid function is usually used for classification when multiple labels are outputted. This challenge is a multi-class classification where one label is output from each file. Softmax function is usually used under this condition. However, the proposed method outputs the results for each model of body part and performs majority voting. We thought that it would be better to list all the classes that could be classified and then perform majority voting. We also tested training with CrossEntropyLoss, which is equiped with a Softmax function. Comparing the testing results, we found that the accuracy was higher when using BCEWithLogitsLoss, so we adopted it.

3.4 Final Prediction Classes Activation

Through the above process, the predictions for 6 sensors with 10 labels of sigmoid values are obtained. Their labels are shown in Table 1. 6 sensors are Right Shoulder, Right Elbow, Right Wrist, Left Shoulder, Left Elbow and Left Wrist. Finally, our method integrates these predictions and outputs the final prediction. Specifically, the sigmoid values for each class are summed and the final prediction is made. If the total values are the same, the class that contains the largest value among each label is adopted.

For example, if the predicted values for activity 1 are [1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.5, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.5, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.5, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3, 0.0, 0.1, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.5, 0.0]. The summed sigmoid values are [1.0, 1.0, 0.5, 0.0, 0.0, 0.0, 0.3, 0.0, 0.6, 0.0], and the label with the largest value is adopted. However, in this case, the maximum value of 1 exists in label 1 and label 2, therefore we go back to the predicted values in each sensor. In the original prediction, the maximum value in label 1 is 1 and the maximum value in label 2 is 0.5, therefore we adopt label 1.

4 Evaluation

This section describes the evaluation experiments. We calculated the accuracy of the proposed model while changing the test subjects in leave-one-subject-out manner.

In partitioning process, repetitions were identified by calculating autocorrelation using `tsa.stattools.acf`[2] in `statsmodels` library. The model is developed in PyTorch. The loss function and optimizer was implemented using PyTorch libraries[3][4].

In the training phase, all data of train subjects were used for training in one epoch, which was iterated 5,000 epochs. Table 3 shows accuracy and loss for the six sensor positions at 5,000 epochs by changing training and testing subjects. The loss was calculated using the 10-dimensional tensor output from the linear layer described in Section 3.2.

Table 3 Accuracy and loss for the six sensor positions at 5,000 epochs by changing training and testing subjects.

Training data	Test data	Sensor position	Accuracy	Loss
Subject 1, 2	Subject 3	Right Shoulder	0.135	0.238
		Right Elbow	0.137	0.266
		Right Wrist	0.127	0.272
		Left Shoulder	0.118	0.265
		Left Elbow	0.119	0.261
		Left Wrist	0.136	0.270
Subject 1, 3	Subject 2	Right Shoulder	0.136	0.250
		Right Elbow	0.121	0.290
		Right Wrist	0.133	0.284
		Left Shoulder	0.097	0.280
		Left Elbow	0.092	0.282
		Left Wrist	0.134	0.279
Subject 2, 3	Subject 1	Right Shoulder	0.142	0.236
		Right Elbow	0.130	0.276
		Right Wrist	0.115	0.282
		Left Shoulder	0.110	0.278
		Left Elbow	0.127	0.267
		Left Wrist	0.110	0.280

From these results, average accuracy of 0.123 was achieved among subjects 1, 2, and 3 in leave-one-subject-out manner. Even if we take into account the fact that there are 10 classes of labels, this accuracy is not high. The reason for the low accuracy could be that some sensor data was missing or mislabeled. As described in Section 2, the training dataset may contain missing data and incorrect activity labels due to the complexity of the motion capture sensor setup. However, we did not take any action on the data with incorrect activity labels. Thus, there is a possibility that training

[2] https://www.statsmodels.org/stable/generated/statsmodels.tsa.stattools.acf.html

[3] https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html

[4] https://pytorch.org/docs/stable/generated/torch.optim.Adam.html

was performed based on the wrong activity labels, and the accuracy was low. In addition, the proposed method uses the autocorrelation function in a preprocessing to extract the repetitive motion. However, it is possible that the data was not extracted correctly because the lengths of the waveforms after the partitioning process were significantly different. It is thought that the accuracy became low due to training on incorrectly extracted data.

Note that for the submitted result for the data of subject 4, our model was trained separately for the body parts with the data of subjects 1, 2, and 3, and the model at 5,000th epoch was used for testing the data of subject 4.

5 Conclusion

This paper reported the solution of our team "RitsBen" to Bento Packaging Activity Recognition Challenge. Our approach calculated the velocity data from the raw data at first. The repetitive motion data was separated from the velocity data by applying an autocorrelation function and extracted seven types of feature values including the mean value. Our approach model consists of 1d convolutional layer, LSTM layer, Linear layer, and Sigmold layer. The model was trained on the data of six parts attached to the upper body separately. The final activation layer, majority voting, was implemented to determine one final prediction label from the six predictions. BCEWithLogitsLoss was used as the loss function. The evaluation results showed that average accuracy of 0.123 were achieved among subjects 1, 2, and 3 in leave-one-subject-out manner. However, the high accuracy could not be achieved due to the fact that we did not deal with the case of incorrectly labeled actions and the possibility that the extraction of repetitive actions was not correct.

References

1. Anguita, D., Ghio, A., Oneto, L., Parra, X., Reyes-Ortiz, J.L.: Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In: International workshop on ambient assisted living, pp. 216–223. Springer (2012)
2. Attal, F., Mohammed, S., Dedabrishvili, M., Chamroukhi, F., Oukhellou, L., Amirat, Y.: Physical human activity recognition using wearable sensors. Sensors **15**(12), 31314–31338 (2015)
3. Bayat, A., Pomplun, M., Tran, D.A.: A study on human activity recognition using accelerometer data from smartphones. Procedia Computer Science **34**, 450–457 (2014)
4. Chen, Y., Xue, Y.: A deep learning approach to human activity recognition based on single accelerometer. In: 2015 IEEE international conference on systems, man, and cybernetics, pp. 1488–1492. IEEE (2015)
5. Dang, L.M., Min, K., Wang, H., Piran, M.J., Lee, C.H., Moon, H.: Sensor-based and vision-based human activity recognition: A comprehensive survey. Pattern Recognition **108**, 107561 (2020)

6. Khan, A.M., Lee, Y.K., Lee, S.Y., Kim, T.S.: A triaxial accelerometer-based physical-activity recognition via augmented-signal features and a hierarchical recognizer. IEEE transactions on information technology in biomedicine **14**(5), 1166–1172 (2010)
7. Lara, O.D., Labrador, M.A.: A survey on human activity recognition using wearable sensors. IEEE communications surveys & tutorials **15**(3), 1192–1209 (2012)
8. Sayeda, S.A., Kohei, A., Nazmun, N., Haru, K., Paula, L., Sozo, I.: Bento packaging activity recognition challenge (2021). DOI 10.21227/cwhs-t440. URL https://dx.doi.org/10.21227/cwhs-t440
9. Tsokov, S., Lazarova, M., Aleksieva-Petrova, A.: An evolutionary approach to the design of convolutional neural networks for human activity recognition. Indian Journal of Computer Science and Engineering **12**(2), 499–517 (2021)
10. Yang, J., Nguyen, M.N., San, P.P., Li, X.L., Krishnaswamy, S.: Deep convolutional neural networks on multichannel time series for human activity recognition. In: Twenty-fourth international joint conference on artificial intelligence (2015)

Appendix

Table 4 Our resources.

Used sensor modalities	Right Shoulder Right Elbow Right Wrist Left Shoulder Left Elbow Left Wrist
Features used	Mean Variance Max Min Root mean square Interquartile range Zero crossing rate
Programming language and libraries used	Python 3.8.8 statsmodels 0.12.2 PyTorch 1.9.0
Window size and post processing	Window size: 400 ms Overlap: 50 ms
Using resources in training and testing	CPU memory: 3043MB GPU memory: 271MB
Training and testing time	Training time (during 5,000 epoch): 1112.233 sec Testing time (at 5,000 epoch): 5.577 sec
Machine specification	OS: Windows 10 Pro CPU: Intel Core i9-10900K 3.70GHz RAM: DDR4 128GB GPU: NVIDIA GeForce RTX 3060 GDDR6 12GB