# Classification of Known and Unknown Environmental Sounds based on Self-organized Space using Recurrent Neural Network

Yang Zhang, Tetsuya Ogata, Shun Nishide, Toru Takahashi, and Hiroshi G. Okuno

*Graduate School of Informatics, Kyoto University, Engineering Building #10, Sakyo, Kyoto 606-8501, Japan*

{*zhang, ogata, nishide, tall, okuno*}*@kuis.kyoto-u.ac.jp*

**Abstract**

Our goal is to develop a system to learn and classify environmental sounds for robots working in the real world. In the real world, two main restrictions pertain in learning. First, robots have to learn using only a small amount of data in a limited time because of hardware restrictions. Second, it has to adapt to unknown data since it is virtually impossible to collect samples of all environmental sounds. We used a neuro-dynamical model to build a prediction and classification system. This neuro-dynamical model can self-organize sound classes into parameters by learning samples. The sound classification space, constructed by these parameters, is structured for the sound generation dynamics, and obtains clusters not only for known classes, but also unknown classes. The proposed system searches on the basis of the sound classification space for classifying. In the experiment, we evaluated the accuracy of classification for both known and unknown sound classes.

*Keywords*: Recurrent Neural Network, Sound Recognition, Neuro-dynamical System

# 1 INTRODUCTION

Recently, there have been a growing number of studies focusing on systems for classifying environmental sounds [1, 2]. Environmental sounds contain a large amount of information, such as those about the dynamic change in the environment. Recognition of environmental sounds is an indispensable ability for creating an autonomous system. Ashikawa developed a model to detect writing movement on a board based on the sound of writing with chalk [3]. Ishihara et al. developed a system to convert environmental sounds into onomatopoeia [4]. Other studies focused on the extracted features of environmental sounds to improve the recognition accuracy of environmental sounds under a fixed framework. They divided samples of environmental sounds from several classes into two, using one for training the model and the other for evaluation [5, 6].

Methods for classifying environmental sounds in previous studies are mainly based on statistical models [7, 8, 9, 10]. Most of these studies show good performances under the condition that environmental sound classes are known (training data is composed of sounds from every sound class considered in the experiment).

The purpose of our study is to develop a system that enables robots working in real world to understand environmental sounds. Such systems require solving of the following two issues.

1. Model should be constructed from a small amount of sound samples as it is difficult to obtain large number of learning sound samples due to durability of hardware.
2. Model should be capable of adapting to unknown sound classes as it is almost impossible to obtain all possible sound samples in advance.

To solve these issues, we apply the following approaches.

1. Creation of the model using neural networks for generalization from small number of training samples.
2. Self-organization of unknown classes in classification space using the generalization capability of neural networks.

We apply recurrent neural network as a dynamical system for the training model of environmental sounds. The dynamical system points out a new possibility for classifying unknown sounds. The concept of the dynamical system is to deal with sequence data by a fixed "rule" generated through training. The model would then infer the "rule" for the recognition and generation processes of unknown sounds. This capability is known as the generalization capability which provides the dynamical system the ability to deal with unknown data using few training data.

Studies have also been conducted to show the capability of dynamical systems to apply to sound classification and generation. Ogata et al. proposed a method to map between different sensory modalities for a robot system to generate motion expressing auditory signals or sounds from the movements of objects [11]. In this study, they associated sounds with motions by using recurrent neural network with parametric bias (RNNPB) [12]. The work showed the capability of the RNNPB to infer unknown sounds from learning samples by generalization and self-organization. Another study focuses on active sensing that exploits the dynamic features of an object [13]. The work trained the RNNPB with data of sounds, arm trajectories, and tactile sensors generated while the robot moved/hit an object with its own arm. The method appropriately configured unknown (untrained) objects in the PB space. Although the objectives of these studies were not classification of environmental sounds, they have shown an insight on how to apply dynamical systems for classifying known and unknown environmental sounds.

Experiments were conducted to detect and classify known/unknown sounds using the constructed model. Detection of unknown sounds was conducted based on the prediction error of the sound. Classification of sounds was conducted using the classification space generated by the model through the training process. The results of the experiment show the effectivity of the model.

The rest of the paper is composed as follows. The overview of the model is described in Section 2. The setup of the experiments with results and discussions are described in Section 3. Conclusion and future work are presented in Section 4.

# 2 ENVIRONMENTAL SOUNDS CLASSIFICATION SYSTEM

In this section, we describe the model to learn and predict environmental sounds.

## 2.1 Multiple Timescale Recurrent Neural Network (MTRNN)

In our model, we utilize the Multiple Timescale Recurrent Neural Network (MTRNN) [14], shown in Figure 1, for the dynamical system. The MTRNN is an extension of the continuous time recurrent neural network which acts as a prediction model to predict the next state as the output, from the current state as the input. The nodes of the MTRNN are composed of input/output nodes ($IO$), fast context nodes ($Cf$), and slow context nodes ($Cs$). The combination weights link nodes in a full connection manner except for those between $IO$ and $Cs$. The main functions of the MTRNN are learning, recognition, and prediction.

In the MTRNN, each node possesses different changing rate controlled by time scale coefficients. More specifically, the fast context ($Cf$) nodes have a high changing rate, which can help to generate dynamics, and the slow context ($Cs$) nodes have a low changing rate, which can help the self-organizing gate to switch the structure of primitive sequence data. The time scale coefficients of $IO$ are the lowest, and $Cs$ are the highest, with the coefficient of $Cf$ set between the two. The function of fast context and slow context are illustrated in Figure 2. During the training process, each primitive sequence is encoded into the initial values of slow context. By selecting an arbitrary initial slow context value, the model can also generate novel primitive sequences.

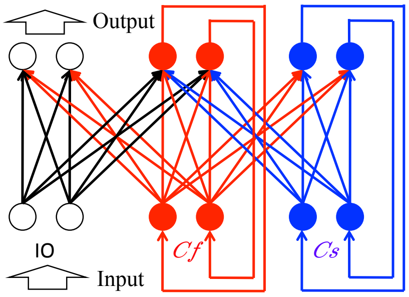


Figure 1: Multiple Timescale Recurrent Neural Network

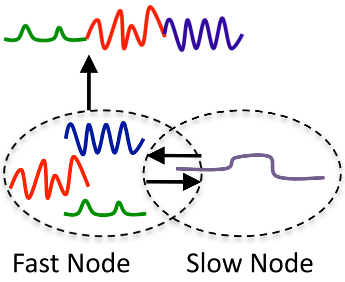


Figure 2: Multiple Timescale Model

The main calculations of the MTRNN are forward calculation and backward calculation (back propagation through time). The three functions of the MTRNN (learning, recognition, and prediction) are conducted using the two calculations. We describe the two calculations in the following. The variable definition used in (1)∼(8) are listed in Table 1.

**Forward calculating step:**

The relation diagram of the variables in the forward calculating step is illustrated in Figure 3. Here, $u_i(t)$ is determined by $x_i(t-1)$ and $u_i(t-1)$, output $y_i(t)$ is determined by $u_i(t)$, and input $x_i(t)$ is determined by $y_i(t)$ and $d_i(t)$.

The forward calculating equations are as follows:

$$\text{if } i \in O \wedge j \in Cs, \text{ or if } i \in Cs \wedge j \in O, \text{ then } w_{ij} = 0.$$

Table 1: Variable Definition

| | | | |
|---|---|---|---|
| $N$ | : number of nodes | $O$ | : $IO$ nodes |
| $d_i(t)$ | : learning data | $\tau_i$ | : time scales |
| $u_i(t)$ | : status | $y_i(t)$ | : output |
| $x_i(t)$ | : input for next step | $w_{ij}$ | : combination weights |
| $\alpha$ | : the learning rate constant | | |

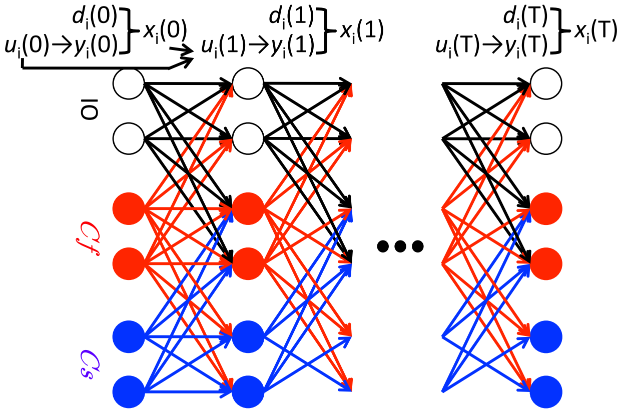


Figure 3: Overview of Forward Calculation

$$u_i(t) = (1 - \frac{1}{\tau_i})u_i(t-1) + \frac{1}{\tau_i}\left[\sum_{j \in N} w_{ij}x_j(t-1)\right] \tag{1}$$

$$y_i(t) = \text{sigmoid}(u_i(t)) \tag{2}$$

$$\text{sigmoid}(x) = \frac{1}{1+\exp(-x)} \tag{3}$$

$$x_i(t) = \begin{cases} \beta \times y_i(t) + (1-\beta) \times d_i(t) & i \in O \\ y_i(t) & \text{otherwise} \end{cases} \tag{4}$$

**Back propagation through time (BPTT) step:**

After forward calculation, the MTRNN conducts BPTT to update the combination weights using the output error (sum square error between learning data and the output obtained through forward calculation). The sum square error is calculated as follows:

$$E = \frac{1}{2}\sum_t \sum_{i \in O}(y_i(t) - d_i(t))^2 \tag{5}$$

The purpose of the BPTT step is to minimize the error function. Partial error differentiation by using combination weights $w_{ij}$ is calculated for minimization as follows:

$$\frac{\partial E}{\partial w_{ij}} = \sum_t \frac{1}{\tau_i}\frac{\partial E}{\partial u_j(t)}x_j(t-1) \tag{6}$$

$\frac{\partial E}{\partial u_j(t)} x_j(t-1)$ is calculated differently for the $IO$ nodes and the $Cf/Cs$ nodes as follows:

$$\frac{\partial E}{\partial u_i(t)} = \begin{cases} (y_i(t) - d_i(t))y_i(t)(1 - y_i(t)) + \\ \left(1 - \frac{1}{\tau_i}\right) \frac{\partial E}{\partial u_i(t+1)} \quad (i \in O) \\ \sum_{j \in N} \frac{\partial E}{\partial u_j(t+1)} [\delta_{ij} \left(1 - \frac{1}{\tau_i}\right) + \\ \frac{1}{\tau_j} w_{ji} y_i(t)(1 - y_i(t))] \; (i \in Cf \; or \; i \in Cs) \end{cases} \tag{7}$$

The following equation is used for updating combination weights:

$$w_{ij}(n+1) = w_{ij}(n) - \alpha \frac{\partial E}{\partial w_{ij}} \tag{8}$$

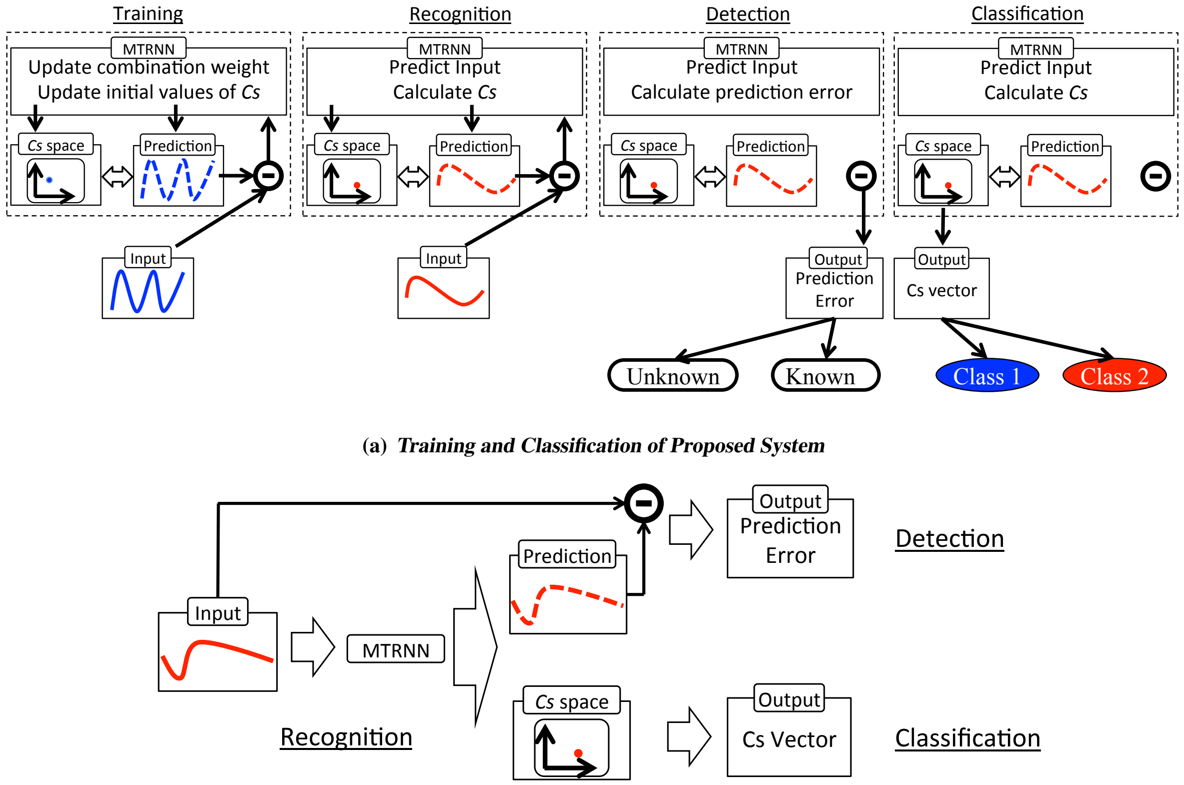
The three main functions of the MTRNN (learning, recognition, and prediction) are conducted as follows.
**Learning:** The MTRNN updates the combination weights and the initial values of *Cs* using training data through forward calculation and BPTT step until the output error converges. In this phase, sequence data used for training are self-organized in the *Cs* space.
**Recognition:** In the recognition function, the MTRNN conducts forward calculation and BPTT as in the learning function. However, during BPTT, the output error is used only to update the initial values of *Cs* (i.e. combination weights are fixed in the BPTT step). Consequently, the process derives one point in the *Cs* space which represents the sequence data to be recognized.
**Prediction:** In the prediction function, the initial values of the input and *Cs* are input into the MTRNN to associate the whole sequence data through forward calculation. As the input of each step, the output of the previous step is directly input into the MTRNN.

## 2.2 Environmental Sound Classification System

The classification system for environmental sounds is illustrated in Figure 4(a). Classification is conducted through four steps of the model (training, recognition, detection, and classification).

### 2.2.1 Detecting unknown sound classes

Detection of unknown sound classes is conducted by recognition and prediction functions of the MTRNN. The recognition function is first used to calculate the *Cs* value representing the sound sequence. The calculated *Cs* is then input into the MTRNN to associate the sound sequence through prediction. The prediction error is calculated by accumulating the absolute errors for each step of the predicted sequence and actual sequence. Prediction errors of unknown sounds are expected to be larger than known (trained) sounds as the MTRNN is not trained with sounds from unknown classes. Therefore, unknown sound classes are classified by comparing the prediction error with a threshold value.



(a) ***Training and Classification of Proposed System***

(b) ***Flow of Proposed System***

Figure 4: Environmental Sound Classification System

### 2.2.2 Classifying sound classes

Classification of sound classes is conducted based on the *Cs* value of the sequence. A detailed flow of classification is shown in Figure 5.

First, several typical known and unknown sounds [1] are selected and input into the MTRNN for recognition to calculate the *Cs* values. These *Cs* values are used as prototypes of nearest neighbor algorithm. Using these prototypes, the sound to be classified is evaluated based on the Euclidean distance between the sound and prototypes. The prototype with the smallest distance is selected and the sound is labeled as the sound class with the selected prototype.

## 3 EXPERIMENTS

In this section, we describe the experiments for evaluating the proposed system.

[1] The unknown sound means no samples of the same class been used to train the MTRNN. Before classifying unknown sounds, the indications of each sounds are calculated in the *Cs* space.

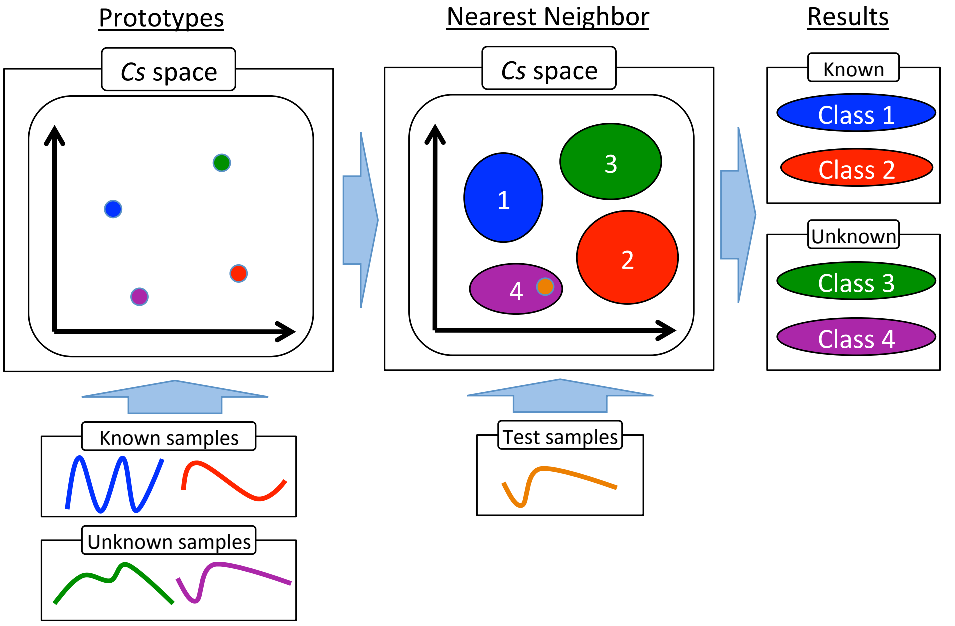


Figure 5: Nearest Neighbor Algorithm for Sound Class Classification

Table 2: Composition of the MTRNN

| | |
|---|---:|
| The number of input nodes | 12 |
| The number of $Cf$ nodes | 20∼40 |
| The number of $Cs$ nodes | 5 |
| The time scale of input nodes | 2 |
| The time scale of $Cf$ nodes | 5 |
| The time scale of $Cs$ nodes | 10,000 |
| Training times | 50,000 |

### 3.1 Condition

In the experiments, we used RWCP real environmental voice sound database to evaluate the performance of our system. From the database, we selected 20 classes of sounds listed in Table 3. Each class is composed of 100 sound data. Four of the 100 data from each class were used for training the model. Before extracting the features, silent parts of the sounds are cut. Then the Mel-frequency cepstral coefficient (MFCC, 12 dimensions) features with a 25-ms window and 10-ms interval were extracted from the sounds. The MFCC features were smoothed and normalized. Relatively long sounds were cut to create MFCC feature sequences with less than 150 steps. We conducted experiments by changing the threshold starting from 0 and increasing by 0.001. We present the result with the best classification performance.

The MTRNN was trained using the MFCC features of training sounds. The composition of the MTRNN is

Table 3: Sound Classes for Experiment

| Class | contents |
|---|---|
| candybwl | Beating a handheld metal box with a metal stick |
| coffmill | Grinding coffee beans with an electric grinder |
| coin1 | Dropping a coin (single) on a wooden board (large) |
| crumple | Crushing copy paper by hand |
| dryer | Sound of hair dryers A and B |
| file | Filing a metal stick with a metal file |
| horn | Blowing a bugle |
| pump | Sound of air pump |
| punch | Punching copy paper with a punch |
| ring | Ringing a bell by shaking |
| saw2 | Sawing a wood piece with a jigsaw |
| shaver | Sound of electric shavers A and B |
| spray | Sound of gas spray A and B |
| stapler | Stapling copy papers with a stapler |
| string | Twanging of a stringed musical instrument |
| tear | Tearing copy paper |
| toy | Sound by releasing spring |
| trashbox | Beating a handheld dustbox with a metal stick |
| whistle1 | Blowing whistle A |
| whistle2 | Blowing whistle B |

shown in Table 2. The process of the experiment is as follows.

1. Divide the twenty classes into four groups as shown in Table 4.
2. Select 11 sets of four-number groups $\{\text{d1}, \text{d2}, \cdots, \text{d11}\}$ randomly.
3. Create inspection cross table constructed by class groups in Table 4 and 11 data groups.
   $\{(\text{c1}, \text{c2}, \text{c3}, \text{c4}), (\text{c1}, \text{c2}), (\text{c1}, \text{c3}), (\text{c1}, \text{c4}), (\text{c2}, \text{c3}), (\text{c2}, \text{c4}), (\text{c3}, \text{c4})\} \times \{\text{d1}, \text{d2}, \cdots, \text{d11}\}$

## 3.2 Result

In this subsection, we present the result of detecting unknown sounds using prediction error and classifying known and unknown sound classes based on the self-organized *Cs* space.

**Detecting unknown sounds using prediction error:**

Table 4: Class Grouping for Cross Validation

| Class Group | Class | Class Group | Class |
|---|---|---|---|
| c1 | candybwl | c2 | coin1 |
| | coffmill | | file |
| | crumple | | pump |
| | dryer | | punch |
| | horn | | ring |
| c3 | saw2 | c4 | shaver |
| | spray | | tear |
| | stapler | | toy |
| | string | | trashbox |
| | whistle1 | | whistle2 |

Figure 6 illustrates the results of average accuracy in detecting unknown sounds based on prediction error. The accuracy of each group is between 63.9% and 82.6%.

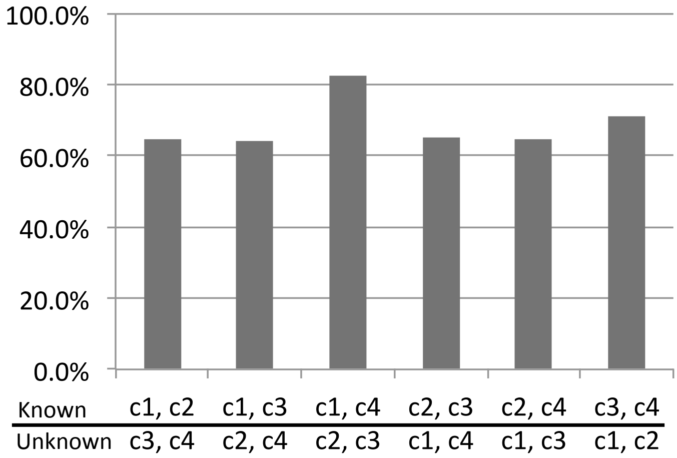


Figure 6: Success Rate of Detection of Unknown Sounds

**The *Cs* space:**

We present several results of the self-organized *Cs* spaces after learning. The *Cs* space with all classes data set (c1, c2, c3, c4 × d1) used for training is shown in Figure 7. The *Cs* space with training data set (c1, c2 × d1), which has the best classification performance is shown in Figure 8. The *Cs* space with training data set (c1, c4 × d1) is shown in Figure 9. (Classes enclosed by the box are known sound classes.)

The *Cs* spaces shown in the figures are the results of principal component analysis (PCA) of the *Cs* values. We present the first two elements of five elements. The accumulated contribution ratio of first two elements of each *Cs* space is also shown in each figure.

From these figures, it is notable that the *Cs* values of each sound forms clusters of the same sound classes, denoting the effectivity of the MTRNN to self-organize sound sequences into the *Cs* space.

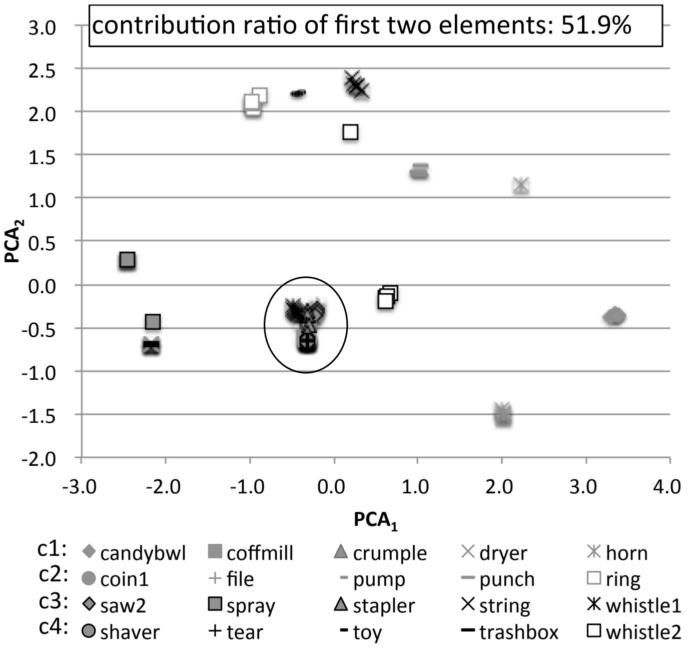


Figure 7: *Cs* Space for Training Set {c1, c2, c3, c4×d1}

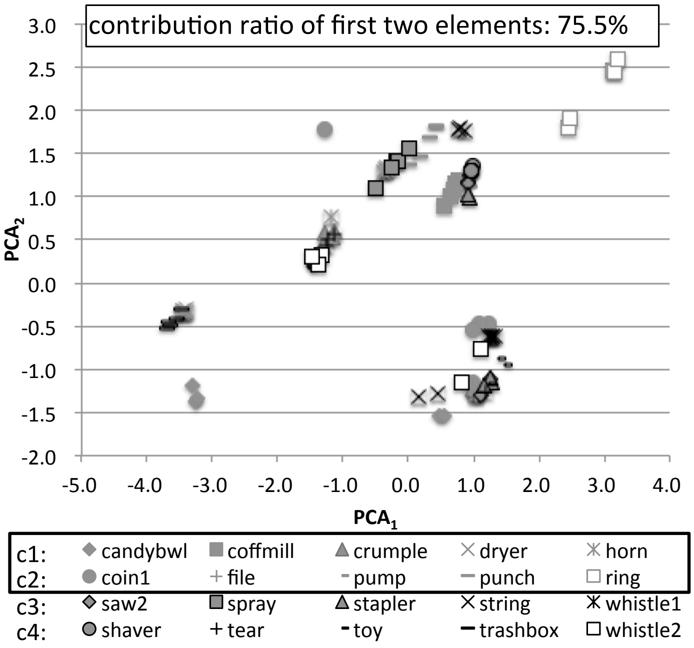


Figure 8: *Cs* Space for Training Set {c1, c2×d1}

**Classifying known and unknown sound classes:**

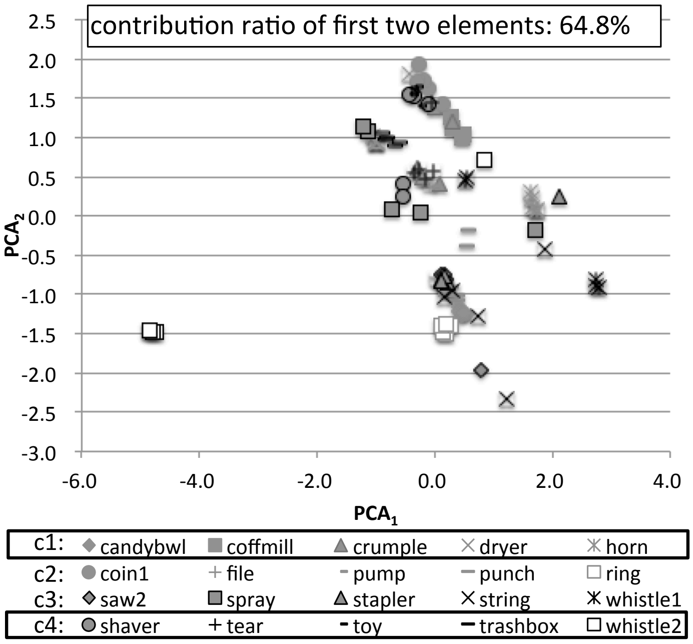


Figure 9: *Cs* Space for Training Set {c1, c4×d1}

Table 5: Success Rate of Classification in Cross Validation

| | Class Group | c1, c2, c3, c3 | c1, c2 | c1, c3 | c1, c4 | c2, c3 | c2, c4 | c3, c4 | Average |
|---|---|---|---|---|---|---|---|---|---|
| | | Success Rate [%] | | | | | | | |
| Known Sounds | Average | 71.0 | 81.2 | 81.8 | 74.8 | 78.2 | 79.6 | 72.4 | 77.0 |
| | Variance | 2.2 | 1.8 | 2.3 | 1.9 | 2.0 | 1.4 | 2.6 | |
| Known and Unknown Sounds | Average | | 67.4 | 71.9 | 67.2 | 73.9 | 70.3 | 67.3 | 69.7 |
| | Variance | | 1.3 | 0.9 | 2.0 | 1.8 | 2.2 | 1.9 | |

Table 5 shows the success rate of classification for each case in cross validation. The row of "Known sounds" in Table 5 shows the average success rate of classification for each group classifying known sound classes. The row of "Known and Unknown Sounds" in Table 5 shows the average success rate of classification for each groups classifying both known and unknown sound classes.

Figure 10 shows the average success rate of classification in different composition of known and unknown classes. From the results, it is notable that the success rate for 10 known classes is the best.

## 3.3 Discussion

In this subsection, we present some discussions considering the results of the experiment.

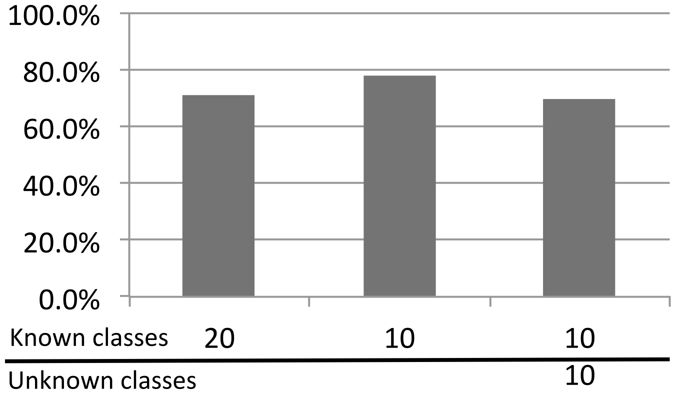


Figure 10: Success Rate of Classification

### 3.3.1 Detecting and classification of unknown sounds

As shown in Figure 6, (c1, c4) class group showed a good performance at detection unknown sounds. The self-organized *Cs* space of (c1, c4) class group learning is shown in Figure 9. Comparing Figure 9 with other *Cs* spaces (such as Figure 8), it is notable that the initial values of *Cs* on this space were not dispersed well. Therefore, it was difficult for (c1, c4) class group to generalize the *Cs* space for unknown sounds. From these results, we suggest that there is a trade-off relationship between classification and the detection of unknown sounds.

### 3.3.2 Relationship between the number of learning data and performance

The success rate of classification for cross validation is shown in Figure 10. In the figure, the 10 known classes group had the best classification performance. The self-organized *Cs* space for learning all classes is shown in Figure 7. From this figure, we can confirm that many classes congest into the circle. Comparing the result with Figure 8 of (c1, c2) class group learning, it is clear that the *Cs* space of (c1, c2) class group is easier for cluster analysis. As prediction errors of (c1, c2, c3, c4) and (c1, c2) were almost the same, the performance of the MTRNN was not affected by the number of samples. Rather, the *Cs* space became more complex for classification analysis requiring more detailed searching in the *Cs* space when training with larger number of samples.

The training result of (c1, c4) class group shows that the initial values of *Cs* on the self-organized *Cs* space were not dispersed as well as other conditions. Compared with the result of (c1, c2) class group in Figure 8, it is notable that the self-organization result is affected by training data.

### 3.3.3 Classifying unknown sounds

From the result of Figure 8 it is notable that unknown sounds are also self-organized in the *Cs* space. The result suggests the possibility of MTRNN to classify unknown sounds without the requirements of prototypes for unknown sound.

# 4 CONCLUSIONS AND FUTURE WORKS

In this paper, we presented a prediction and classification system for environmental sounds using a neural-dynamical model. This system showed a new approach for classifying unknown environmental sounds using a small amount of samples.

For the evaluation experiment, we selected 20 classes from RWCP real environmental voice sound database, and trained the system using sequence data of Mel-frequency cepstrum coefficient (MFCC) features extracted from sounds. Four data samples out of 100 for each class were used for training. The experiment was conducted with 10 known (trained) classes and 10 unknown classes. The classification performance of the model by cross validation. The success rate of classification for known classes was 77.0%, and that of unknown classes was 69.7%. The results show the effectivity of the system to deal with both known and unknown sound classes.

To develop the system to apply to actual robotic platform, we plan to improve the model as follows. First, we plan to evaluate the effectivity of the model in comparison to other methods, such as GMM. Second, we plan to design a selection technique that effectively determines the threshold value for detecting unknown sounds. Third, we plan to develop an effective algorithm for clustering the non-linear *Cs* space into sound classes. Finally, we plan to design a method to create clusters of unknown sound classes automatically without the requirements prototypes. We hope our work would contribute to robotic systems to recognize and classify environmental sounds.

# ACKNOWLEDGMENTS

This research was supported by JST PRESTO (Information Environment and Humans), Grant-in-Aid for Creative Scientific Research (19GS0208), and Grant-in-Aid for Scientific Research (B) (21300076).

# REFERENCES

[1] T. Ashiya and M. Nakagawa, "A proposal of a recognition system for the species of birds receiving birdcalls : An application of recognition systems for environmental sound," *IEICE transactions on fundamentals of electronics, communications and computer sciences*, vol. 76, no. 10, pp. 1858–1860, 1993.

[2] V. Exadaktylos, M. Silva, J.-M. Aerts, C. Taylor, and D. Berckmans, "Real-time recognition of sick pig cough sounds," *Computers and Electronics in Agriculture*, vol. 63, no. 2, pp. 207–214, 2008.

[3] T. Ashikawa, A. Suganuma, and R. Taniguchi, "Development and validation of an automatic camera control system made with a detection of a chalking sound," *IEICE. ET*, vol. 102, no. 509, pp. 43–48, 2002.

[4] K. Ishihara, K. Komatani, T. Ogata, and H. G. Okuno, "Sound-imitation word recognition for environmental sounds," *Transactions of the Japanese Society for Artificial Intelligence*, vol. 20, no. 3, pp. 229–236, 2005.

[5] M. Cowling and R. Sitte, "Comparison of techniques for environmental sound recognition," *Pattern Recognition Letters*, vol. 24, no. 15, pp. 2895–2907, 2003.

[6] N. Yamakawa, T. Kitahara, T. Takahashi, K. Komatani, T. Ogata, and H. G. Okuno, "Effects of modelling within- and between-frame temporal variations in power spectra on non-verbal sound recognition," *Proceedings of International Conference on Spoken Language Processing (Interspeech 2010)*, pp. 2342–2345, 2010.

[7] S. Shimura, Y. Hirano, S. Kajita, and K. Mase, "Experience movie presentation method using action situation query," *IPSJ*, pp. 4_81–4_82, March 2006.

[8] Y. Sasaki, M. Kaneyoshi, S. Kagami, H. Mizoguchi, and T. Enomoto, "Daily sound recognition using pitch-cluster-maps for mobile robot audition," in *IROS'09: Proceedings of the 2009 IEEE/RSJ international conference on intelligent robots and systems*. Piscataway, NJ, USA: IEEE Press, 2009, pp. 2724–2729.

[9] K. Miki, T. Nishimura, S. Nakamura, and K. Shikano, "Environmental sound discrimination based on hidden markov model," *Information Processing Society of Japan SIG Notes*, vol. 99, no. 108, pp. 79–84, 1999.

[10] K. Nakamura, R. Nisimura, A. Lee, H. Saruwatari, and K. Shikano, "Identification of environmental noise and unnecessar utterance on a real information guidance system with spoken dialogue interface," *IEICE. SP*, vol. 103, no. 632, pp. 13–18, 2004.

[11] T. Ogata, S. Nishide, H. Kozima, K. Komatani, and H. G. Okuno, "Inter-modality mapping in robot with recurrent neural network," *Pattern Recognition Letters*, vol. 31, no. 12, pp. 1560–1569, 2010.

[12] J. Tani and M. Ito, "Self-organization of behavioral primitives as multiple attractor dynamics: A robot experiment," *IEEE Transactions on Systems, Man, and Cybernetics A*, vol. 33, no. 4, pp. 481–488, 2003.

[13] T. Ogata, H. Ohba, K. Komatani, J. Tani, and H. G. Okuno, "Extracting multimodal dynamics of objects using rnnpb," *Journal of Robotics and Mechatronics, Special Issue on Human Modeling in Robotics*, vol. 17, no. 6, pp. 681–688, 12 2005.

[14] Y. Yamashita and J. Tani, "Emergence of functional hierarchy in a multiple timescale neural network model: A humanoid robot experiment," *PLoS Comput Biol*, vol. 4, no. 11, p. e1000220, 11 2008.