Proc. 2019 IEEE Int. Conf. on Advanced Robotics and Mechatronics (ARM 2019), Osaka, Japan, July 2019.

# An Incremental Feature Set Refinement in a Programming by Demonstration Scenario

Hoai Luu-Duc and Jun Miura
Department of Computer Science and Engineering
Toyohashi University of Technology
Email: hoai@aisl.cs.tut.ac.jp, jun.miura@tut.jp

***Abstract*— In transferring knowledge from human to robot using Programming by Demonstration (PbD), choosing features which can represent the instructor demonstrations is an essential part of robot learning. With a relevant set of features, the robot can not only have a better performance but also decrease the learning cost. In this work, the feature selection method is proposed to help the robot determine which subset of the features is relevant to represent a task in PbD framework. We implement an experimental PbD system for a simple task as proofing our concept as well as showing the preliminary results.**

***Index Terms*— Human-Robot Interaction, Programming by Demonstration, Feature Selection**

## I. INTRODUCTION

Transferring knowledge is defined as an activity that an instructor shares or disseminates knowledge to a learner. In robotic field, human-robot knowledge transfer is one of the areas that attracts attention of mamy researchers. There are two parts in the human-robot knowledge transfer. The first part is the human-to-robot part in which a robot learns knowledge from a human through human instruction and the other part is the robot-to-human part where a robot teaches the task to a human [1]. To transfer knowledge from an instructor to a learner, Programing by Demonstration (PbD) is one of the approaches that enables the instructor can able to share the knowledge to the leaner by demosntration a senquence of examples [2].

The task in our work is defined as follows. The world is characterized by its state and a robotic task is specified its desired state (goal state). From a sequence of demonstrations, the robot repeatedly observes the current state and selects an action based on its observation, to transfer the current state to the next one, and eventually reaches the final state (i.e. achieving the task). Based on this definition, if each state is well defined by a set of selected features, the robot can properly understand the task and easily select an appropriate action. However, the number of possible features is usually large due to a complex nature of the real world, and therefore selecting an appropriate set of features is difficult from a limited amount of demonstrations. As a result, the selected set if features might include irrelevant ones which not only do not represent the state of the task but also lead to poor performance in transferring knowledge. Therefore, in this paper, a feature selection approach is proposed to help the robot evaluate and select a relevant subset of features which can represent the task.

Feature selection in robotics has been applied to several problems. Deuk et al. [3] used a feature selection to solve a mobile robot navigation problem. Loscalzo et al. [4] developed a feature selection method for a genetic policy search. Kim et at. [5] applied a feature selection to a rescue robot to classify smoke and fire. Bullard et al. [6] enabled the robot to interact with a human requiring the features information. However, these works assume that demonstrations or examples are provided completely while demonstrations in PbD are limited and depend on a instructor. If a instructor can provide a good demonstration subset, a robot can have a good performance with a small set of demonstrations. In contrast, with a bad demonstration subset, the robot needs more demonstration to understand the task or has a low performance in executing the task. Feature selection is thus one of the promising approaches to cope with a limited amount of demonstrations by generating a relevant feature subset.

In this work, we propose a feature selection method to generate a promising subset of features incrementally after each demonstration. The advantage of our approach is that we do not need to wait for a complete set of demonstrations. The proposed method generates a promising subset after each demonstration then the robot will use it with the current demonstrations set to create a model of the task and execute it under the instructor observation. The main contribution of the paper is to propose a feature selection framework that can generate an appropriate set of feature subset from a limited demonstrations set.

The rest of the paper is organized as follows. Section 2 presents the learning mechanism in which the feature selection method is proposed to help the robot refine the relevant subset of features. Section 3 shows the experimental results and Section 4 presents concluding remarks and future work.

## II. FEATURE SELECTION FOR TASK MODELING

### *A. Problem Statement*

The problem is to determine a promising subset of features which is used for describing the task. To define the state, the robot is given a set of demonstrations with the binary label, $L$, and the list of all candidate features, $F$, among which the robot extracts promising ones based on the demonstration. The goal of feature selection is to define which subset of

TABLE I: The list of features in Simulation

| Object | Features | Values |
|---|---|---|
| Blocks | background color | red, green |
| | bounding color | red, green |
| | outside shape | cubic, cylinder |
| | inside shape | cubic, cylinder |
| | size | 1,2 |
| | redundant features [f1-f10] | 0,1 |
| Place | background color | red, green |
| | bounding color | red, green |
| | outside shape | rectangle, ellipse |
| | inside shape | rectangle, ellipse |
| | size | 1,2 |
| | redundant features [f1-f10] | 0,1 |

the features, $F'$, represents the set of demonstrations and then is used to classify the demonstration to create the task model. In this research, we choose a simple task which has only one action to teach the robot. In other words, a task is represented by the target state to achieve. We assume that the robot already has a planning and execution system that can transfer one state to another.

### B. Problem Domain

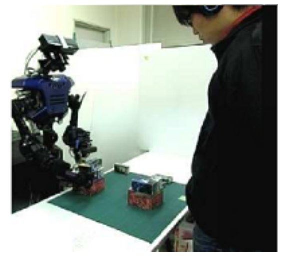
In this work, we give the robot a pick-and-place task. The instructor demonstrates a task by putting an object to an appropriate place under specific rules. For example, the color of the object must be the same with that of the place. The robot can observe three types of attributes (color, shape and size) of objects and places. The label of demonstration is a binary label; zero means a false state and one means a true state. There are 30 features (shown in Table I) that can be observed by the robot. Redundant features are added to increase the complexity and the noise of the learning task. The goal of feature selection is to determine which subset of features is relevant to the demonstration subset.

### C. Learning from Demonstration Process

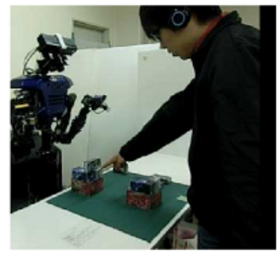
The Learning from Demonstration is conducted by the following steps:

1) Demonstration by instructor: the instructor demonstrates the task by picking an object then putting it into an appropriate location. The definition of the task is given to the instructor before teaching. For example, the blue object must be put into the red place as shown in Fig. 1a.
2) Observation and feature selection by the robot: the robot observes the demonstration, then extracts all information from demonstration (i.e, the features and their values). Then, the feature selection algorithm is applied to calculate which subset of features is relevant to the task and the robot chooses the highest relevant subset of feature as a model.
3) Task execution by the robot: using the model in step 2, the environment of the task is refreshed then the robot executes the task under the supervision of the instructor as shown in Fig. 1b.

(a) Instructor's demonstration



(b) Robot's execution



(c) Instructor's judgment

Fig. 1: Programing by Demonstation Framework [1]

4) Judgment: after observing the robot's execution, the instructor judges if the action of the robot is correct as shown in Fig. 1c. If the robot fails to execute the task correctly, the new demonstration will be presented, go to step 1. If the robot succeeds in the task, the learning process will finish.

### D. Learning Framework

*1) Feature Selection Approaches:* There are the following three main methods in feature selection: filter method, wrapper method and embedded method. Filter methods use variety technique as a ranking criteria for feature selection. The ranking criteria is used to score features and a threshold is pre-defined to remove features below the threshold. Wrapper methods use the classifier performance as the objective function to evaluate feature subsets. The feature subsets are generated by employing a search algorithms. Embedded methods try to decrease the computation cost of reclassifying different feature subsets which is done in wrapper methods by incorporating feature selection as a part of the training process. LASSO regession is one of the approaches of this method. Bullard [6] had shown that filter methods are more efficient in generating promising feature subset in terms of the computation cost and accuracy. Filter methods score features under specific criteria and the top highest score features are chosen as a promising feature subset. However, with a limited demonstrations, there are a variety of feature subset that can represent current demonstrations. As a result, the feature selection might fail to get a promising feature subset. For this reason, we use a wrapper method: our proposed framework tries to add as many features as possible to a promising feature subset which can represent the current demonstrations using a mutual information criteria, and then to remove irrelevant features from the promising subset by

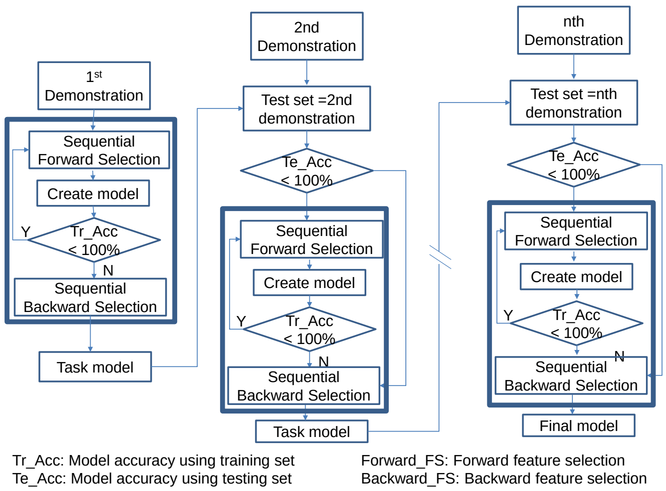



Fig. 2: Learning Framework using Programming by Demonstration

a redundancy analysis before generating the final model of the task.

*2) The Proposed Learning Framework:* Figure 2 shows the learning process. After demonstrating the first demonstration, the Sequential Forward Selection (SFS) algorithm [7] generates a promising subset of features, then the model (i.e. classification) is created based on the subset and the current demonstration set. In our work, the ID3 Decision Tree algorithm [10] is used as the classifier. After the model creation step, the training accuracy is calculated to determine if the current promising features subset can represent the current demonstration set. If the training accuracy is equal to 100 percent, the Sequential Backward Selection (SBS) algoritm [7] is applied to remove irrelevant features to create the task model and the robot wait for the next demonstration.

In the second demonstration, after observing the instructor's demonstration, the robot will calculate the testing accuracy of the model in the last demonstration with the new demonstration. If the testing accuracy high enough (in this case, it equal to 100 percent, the SBS is applied to remove irrelevant features . If not, the SFS will add more features into the previous promising features subset and create a new model of the task. The learning process will finish when both the testing accuracy and the training accuracy are equal to 100 percent.

*a) Sequential Forward Selection:* To add the features to a promising features subset, Mutual Information (MI) $I(X,Y)$ (as described in Eq. (1)) is used to measure the amount by which the class $Y$ uncertainty is reduced after having observation from variable $X$.

$$I(X,Y)=\sum_{x\in X}\sum_{y\in Y}p(x,y).\log_2\frac{p(x,y)}{p(x).p(y)} \tag{1}$$

The main goal of feature selection is to select a smallest number of features that carry as much information as possible. So the feature selection problem can be formed as follows: given the features list $X=\{x_i : i\in A=\{1,...,n\}\}$, the goal is to find the subset $X_S=\{x_j : j\in S\subset A\}$ which maximizes the mutual information $I(S,Y)$. However, computing the MI between the class and all candidate features subset is impossible, so the Conditional Mutual Information (CMI) [8] (Eq. (2)) is applied to decrease the computation cost for ranking the features.

$$I(x_i,Y|X_S)=\sum_{X_S}p(X_S)\sum_{x_i\in X}\sum_{y\in Y}p(x_i,y|X_S)\log_2\frac{p(x_i,y|X_S)}{p(x|X_S).p(y|X_S)} \tag{2}$$

where $X_i$ is a feature in list of features that can be observed by a robot, $Y$ is a label list of demonstrations and $X_S$ is a promising features subset.

Using the CMI, the reducing uncertainty between class $Y$ and feature $X_i$ under an observation of subset $X_S$ is calculated. If the CMI is equal to zero, there is no information under the observation by adding the feature $X_i$, that means $X_i$ does not provide any information to predict $Y$ while other features in subset $S$ are known. By this way, the CMI can find a promising subset without testing all of candidate features subsets.

To directly calculate the CMI, the complex joint probability of subset $X_S$ must be computed. This computation is very expensive when the number of feature in subset $X_S$ increase. To solve this problem, Wang et al. [9] proposed The Conditional Mutual Information Maximin Algorithm (CMIM) to deal with problem. In the CMIM algorithms, instead of calculating the joint probability of $X_S$, the conditional mutual information between every features in the features list and the label for each feature in a promising subset $X_S$ is computed. Then, the next feature that can be added to a promising subset must satisfy:

$$X_{new}=max(min(I(X_i,Y|X_j)) \tag{3}$$

The algorithms of SFS is shown in algorithm 1. If there is no feature in a promising features subset $S$, the MI is calculated between each features in features list and the label of demonstration to add the first feature into $S$. Otherwise, the conditional mutual information is calculated and the next feature which is added into $S$ is chosen based on Eq. (3).

*b) Creating Task Model:* After finishing Forward Feature Selection, the promising feature subset and the current demonstration set is used for training and creating the task model. After training process, the training accuracy is calculated to determine if the current features subset can present the latest demonstration set. If it is not, more features will be added into the current subset by SFS.

*c) Sequential Backward Selection:* If the task model passed the testing accuracy criterion, the SBS module is executed to remove irrelevant features in the subset. The main reason to have this module is that the demonstration is limited while the number of features in the scene is very large, and that lead to the irrelevant features might join into the promising features subset. To remove irrelevant features, Symmetrical Uncertainty (SU) [11] which is defined in Eq. (4) is used to estimate the redundancy value of each feature

**Algorithm 1** The Sequential Forward Selection algorithm

**Input:** Demonstration
**Output:** Promising Feature Subset $S$
extract $featureslist$ and data from demonstration
$listMI = NULL$
**if** $S$ == $NULL$ **then**
  **for** *feature in features list* **do**
    $listMI$.append(calculate MI($feature$,$label$))
  **end**
  $S$.append($feature$ which have highest score)
**else**
  features which are in $S$ are removed from $featurelist$
  **for** *x feature in features list* **do**
    $tempMI = NULL$
    **for** *s feature in S* **do**
      $tempMI$.append(the conditional MI( $xfeature$ , $sfeature$ ))
    **end**
    $listMI$.append(min($tempMI$))
  **end**
  $S$.append(argmax($listMI$))
**end**
return $S$

in the current subset. The symmetrical uncertainty is defined as:

$$SU(X_i,Y) = 2\frac{IG(X_i|Y)}{H(X_i)+H(Y)}, \tag{4}$$

where $H(X_i)$ and $H(Y)$ are the entropy of features in the feature list and label data respectively. $IG(X_i|Y)$ that is given by Eq. (4) is called information gain [12], defined as:

$$IG(X|Y) = H(X) - H(X|Y). \tag{5}$$

To analyse the redundancy of a feature, C-correlation and F-correlation [7] are defined. C-correlation is the correlation between any feature $X_i$ and the label $Y$, denoted by $SU_{i,y}$. F-correlation is the correlation between any pair of features $X_i$ and $X_j$ $(i \neq j)$, denoted by $SU_{i,j}$. If $SU_{j,y} \geq SU_{i,y}$ and $SU_{i,j} > SU_{i,y}$, $X_j$ forms an approximate Markov blanket for $X_i$. In this case, a relevant feature is called predominant feature if it does not have any approximate Markov blanket in the current set.

The SBS algorithm is shown in algorithm 2. In the first step, the SU value between each feature in a promising feature subset and label is calculated, then all of the features are ordered in a descending order according to their SU values. In the next step, F-correlation among each feature pair is calculated and predominant features are identified using approximate Markov blanket definition. After getting the list of irrelevant features, each feature in the list will be removed if it does not change the training accuracy. The training accuracy is used in this step because the demonstration is limited and the final model is required to have 100 percent in training accuracy. The final task model is the model that has the highest training accuracy and the lowest number of features in the promising feature subset.

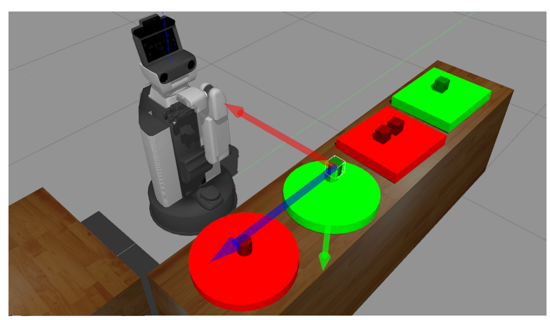
**Algorithm 2** The Sequential Backward Selection algorithm

**Input:** Demonstration, Promising Features Subset $S$
**Output:** Removing Feature Subset $Sr$
**for** $feature_i$ *in S* **do**
  calculate $SUi,y$ for $feature_i$
**end**
sort $S$ in descending $SUi,y$ value
**for** $feature_j$ *in S* **do**
  **for** $feature_i$ $(i <> j)$ *in S* **do**
    calculate $SUi,j$ for $feature_j$ and $feature_i$
    **if** $SUi,j > SUi,y$ **then**
      $Sr$.append($feature_i$)
      $S$.remove($feature_i$)
    **end**
  **end**
**end**
return $Sr$

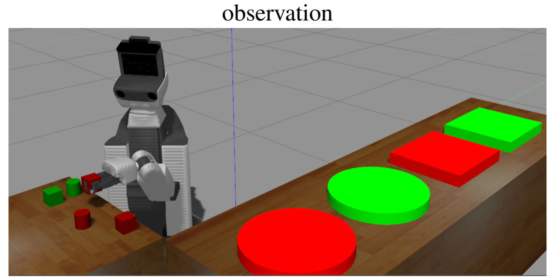
## III. EXPERIMENT RESULTS

In our experiment, the task that the robot needs to learn is the matching task, for example, the background color of the object must be the same with that of the place. The rule of matching depends on the instructor. The goal of the robot is to find the features subset which can represent the rule of the task and execute it correctly. To test the proposed method, a simple simulator is created to let the instructor demonstrate the task as shown in Fig. 3

In this simulator, the places that the block must be put are on the right side while the blocks are on the left side. To analyze easily the subset of features, we use only two type of color (red and green) and two type of shape (cubic and cylinder) in this experiment. The instructor uses a mouse

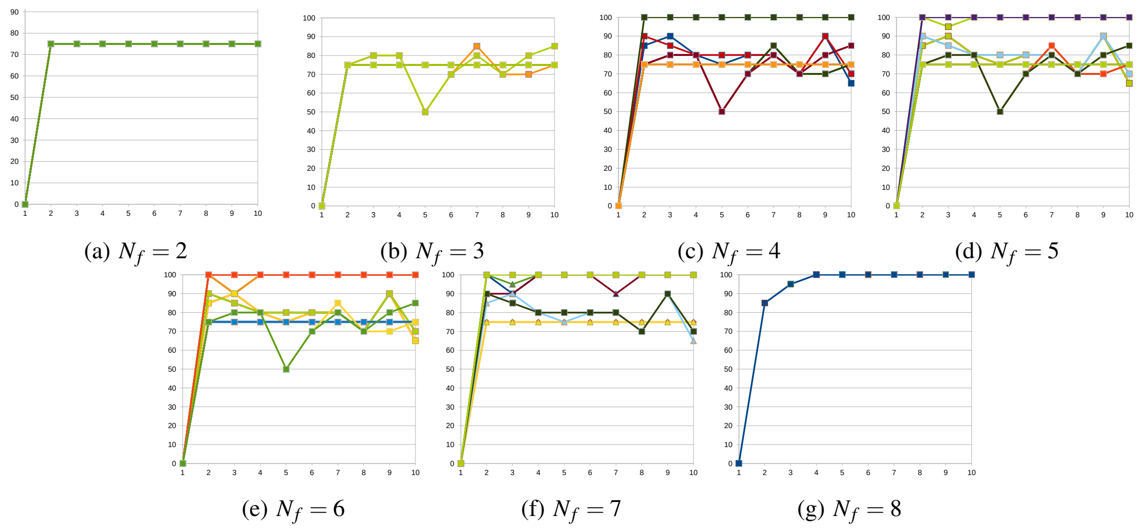


(a) Instructor's demonstration under robot's observation



(b) Executing the task after observation

Fig. 3: The simulator for Task Learning



(a) $N_f = 2$ (b) $N_f = 3$ (c) $N_f = 4$ (d) $N_f = 5$

(e) $N_f = 6$ (f) $N_f = 7$ (g) $N_f = 8$

Fig. 4: The accuracy of model with each feature subset

or a joystick to move a block to an appropriate location in demonstrating the task under specific rules (Fig. 3a). After moving all the blocks to its location, the instructor uses the finish button to inform the robot that the demonstration has finished. The blocks in each demonstration are randomly created and the instructor may not be able to provide enough instances in one demonstration. In each demonstration, five blocks are created to be put in the corresponding place. Moreover, when a block is put in an appropriate place, one positive data and three negative data are created. The positive data is the matching between this block and its place while negative data are the matching between this block and the other places. In this task, there are totally 16 instances which are necessary to represent the goal state. After each demonstration, the robot will show the promising features subset to the instructor and execute a task under an instructor's observation (Fig. 3b). The learning process will finish when the promising feature subset is the same with the instructor rule or the demonstration set is enough.

Figure 4 shows the relationship between the accuracy of the model and the number of features in the model. In this experiment, we used eight features. All of possible combination of features as subsets are generated and their accuracies are calculated. The number of true features is four. From the chart, if the number of features is not sufficient ($N_f < 4$), the accuracy of the model cannot reach 100 percent accuracy, that is, the model cannot represent a task correctly. In contrast, if the number of features in features subset is large, that is redundant features are in a feature subset, that could lead to overfitting problem. So that, choosing the number of features to represent a task is essential part in feature selection.

In the next experiment, to test the efficiency of the proposed framework, 5 different tasks, for which the true number of features is 4 features, are demonstrated. Fig 5 shows the testing accuracy of each model. Each task have 11 demonstrations. At the end of the demonstration, every

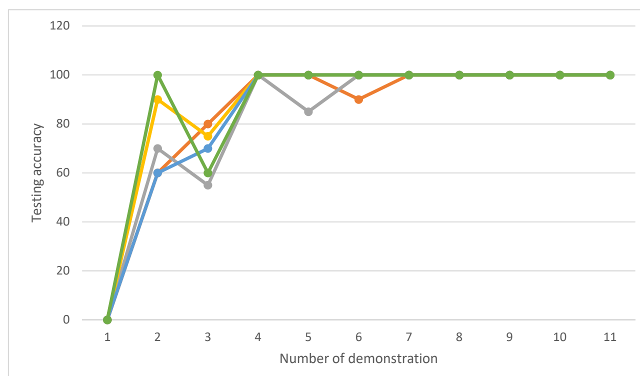



Fig. 5: The 5 tasks models accuracy

task can reach 100 percent in the testing accuracy. However, there are 2 tasks that have a promising features subset which is different from the true one. The reason for this situation is that demonstrations is limited and it is depends on an instructor. So the greedy search approach might fail in scoring features in the features list. Moreover, 5 tasks include complementary features (for example the color of the object must be the same with the color of the place), while the CMIM algorithms cannot solve this problem [13]. This problem must be considered to improve the criteria in the sequential forward selection.

To analyze the effect of the demonstration set to the learning performance, we implement two types of demonstrations for the matching task. One is a random demonstration generation in which the instructor just performs random demonstrations that are generated by the simulator. The other is a deliberate or careful demonstration where the instructor carefully chooses demonstrations. Table II shows

TABLE II: Experiment Results

| Learning Type and Number of Demonstrations | True Features Set | The Promising Feature Set |
|---|---|---|
| Random generation demonstration (24 demonstrations) | Object background color<br>Object outside shape<br>Place background color<br>Place outside shape | Object background color<br>Object inside shape<br>Place f2<br>Place inside shape |
| Deliberate demonstration (4 demonstrations) | Object background color<br>Object outside shape<br>Place background color<br>Place outside shape | Object background color<br>Object outside shape<br>Place background color<br>Place outside shape |

some results of learning a matching task. In the random demonstration generation, the robot failed in choosing the true feature set because the robot use the greedy search based on the Mutual Information and the conditional Mutual Information to choose the relevance features. So with a limited number of demonstrations, there are a variety of results that match with the current demonstration set. In this case, the instructor stops the learning process after 24 demonstrations because the demonstration set is completed. In the deliberate or careful demonstration learning, the instructor deliberately chooses the demonstrations which can represent the task easily without duplication. In this case, the robot can reach the true features set after 4 demonstrations and execute the task correctly.

## IV. CONCLUSIONS AND FUTURE WORK

In this paper, a feature selection method to help the robot achieve the task is proposed. By adding demonstration and using a feature selection during the learning process, the robot can choose the relevant features and execute the task correctly. However, the promising feature subset depends on the demonstration set, that is, if the robot has a good set of demonstrations, the robot can get the correct feature subset easily. Otherwise, the robot might take a long time in learning to refine the features subset.

Using only feature selection method is not enough to refine the features subset which represent the task. As future work, the human-robot interaction must be considered to improve the accuracy of the system. For example, the robot may ask the instructor about the features information or acquire the demonstration from the instructor during the learning process. Finally, the complementary features must be considered in future.

## ACKNOWLEDGMENT

The authors would like to thank Toyota Motor Corporation for providing software of their Human Support Robot (HSR) [14]. This work is in part supported by JSPS KAKENHI Grant Number 17H05860.

## REFERENCES

[1] K. Yamada and J. Miura, “Ambiguity-Driven Interaction in Robot-to-Human Teaching”, *Proceedings of the Fourth International Conference on Human Agent Interaction*, 2016.

[2] B.D. Argall, “A survey of robot learning from demonstration”, *Robotics and Autonomous Systems (2009)*, doi:10.1016/j.robot.2008.10.024

[3] C. Diuk, L. Li, and B. R. Leffler, “The adaptive k-meteorologists problem and its application to structure learning and feature selection in reinforcement learning”, *International Conference on Machine Learning*, 2009.

[4] S. Loscalzo, R. Wright, and L. Yu, “Predictive feature selection for genetic policy search”, *Autonomous Agents and Multi-Agent Systems*, 2015.

[5] J.-H. Kim, S. Jo, and B. Y. Lattimer, “Feature selection for intelligent firefighting robot classification of fire, smoke, and thermal reflections using thermal infrared images”, *Journal of Sensors*, 2016.

[6] K. Bullard, S. Chernova1, and A.L. Thomaz, “Human-Driven Feature Selection for a Robot Learning Classification Tasks from Demonstration”, *Robotics and Automation (ICRA), 2018 IEEE International Conference on. IEEE*, 2018.

[7] L. Yu, and H. Liu, “Efficient feature selection via analysis of relevance and redundancy”, *Journal of machine learning research 5.Oct (2004)*, 2004.

[8] J. Novovičová, et al, “Conditional mutual information based feature selection for classification task”, *Iberoamerican Congress on Pattern Recognition. Springer, Berlin, Heidelberg*, 2007.

[9] G.Wang, et al, “Feature selection with conditional mutual information maximin in text categorization”, *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, 2004.

[10] J. R. Quinlan, “Induction of Decision Trees”, Mar. 1986.

[11] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, “Numerical Recipes in C.” *Cambridge University Press, Cambridge*, 1988.

[12] J. R. Quinlan, “C4.5: Programs for Machine Learning.” *Morgan Kaufmann*, 1993.

[13] P. E. Meyer, C. Schretter, and G. Bontempi, “Information-theoretic feature selection in microarray data using variable complementarity”, *IEEE Journal of Selected Topics in Signal Processing*, 2008.

[14] T. Yamamoto, et al, “Development of Human Support Robot as the research platform of a domestic mobile manipulator.” *ROBOMECH Journal 6.1*, 2019