2022 International Conference on Rehabilitation Robotics (ICORR)
25-29 July, 2022, Rotterdam, The Netherlands

2022 International Conference on Rehabilitation Robotics (ICORR) | 978-1-6654-8829-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICORR55369.2022.9896498

Learning Dynamic Patient-Robot Task Assignment and Scheduling for A Robotic Rehabilitation Gym

Bikranta Adhikari, Shivanjali Ranashing, Benjamin A. Miller, Vesna D. Novak, *Senior Member, IEEE*, and Chao Jiang, *Member, IEEE*

Abstract— **A robotic rehabilitation gym is a setup that allows multiple patients to exercise together using multiple robots. The effectiveness of training in such a group setting could be increased by dynamically assigning patients to specific robots. In this simulation study, we develop an automated system that dynamically makes patient-robot assignments based on measured patient performance to achieve optimal group rehabilitation outcome. To solve the dynamic assignment problem, we propose an approach that uses a neural network classifier to predict the assignment priority between two patients for a specific robot given their task success rate on that robot. The priority classifier is trained using assignment demonstrations provided by a domain expert. In the absence of real human data from a robotic gym, we develop a robotic gym simulator and create a synthetic dataset for training the classifier. The simulation results show that our approach makes effective assignments that yield comparable patient training outcomes to those obtained by the domain expert.**

I. Introduction

A. Rehabilitation Robotics and Robotic Gym

While rehabilitation robots have mostly been utilized in a setup consisting of one patient, one robot, and one therapist, the last decade has seen an increase in multi-robot systems: setups where two or more patients can exercise together using two or more robots while supervised by a single therapist [1]–[3]. A setup specifically involving three or more rehabilitation robots (or other rehabilitation devices such as passive limb trackers) can be referred to as a robotic rehabilitation gym [4]. While such robotic rehabilitation gyms have not yet seen extensive evaluation, results so far indicate that rehabilitation outcome is comparable to one-on-one therapy [4], [5], with the additional advantage that a single therapist can supervise more than one patient. A recent observational study suggests that a therapist could effectively supervise up to four patients in a robotic gym [6], reducing the need for expert manpower in rehabilitation.

In a robotic gym, patients would ideally move between robots within a session as they, e.g., improve their performance or get tired of an exercise. However, in existing studies, patients either did not switch robots within the session or switched between them arbitrarily after a predetermined time period [4], [5]. Thus, the effectiveness of a robotic gym could potentially be increased further by dynamically assigning patients to specific robots. While this could be done by a therapist, it could also be done by a central artificial intelligence that monitors each patient's behavior and suggests when they may benefit from switching robots. Such an intelligent patient monitoring and dynamic patient-robot assignment system could assist a human therapist or operate with full autonomy to achieve optimal group rehabilitation outcome.

B. Dynamic Task Allocation and Scheduling

Dynamic patient-robot assignment in a robotic gym can be viewed as a task allocation and scheduling problem where a set of tasks are assigned to a set of resources or agents such that an overall performance objective is achieved [7]. Task allocation and scheduling problems have been studied in a variety of application domains, including human-robot collaborative assembly [8], daily assistance for elderly [9], medical services [10], and military operations [11]. However, they have not yet been examined in the context of group rehabilitation, where the domain features and requirements may be different.

Task allocation and scheduling is commonly formulated as a mathematical programming problem such as mixed-integer linear programming (MILP) [12] and constraint programming (CP) [13]. Optimization-based approaches are leveraged to obtain either an exact or approximate solution depending on the complexity of the specific problem. The computation of an exact solution usually becomes intractable for complex problem domains. A common practice is to apply heuristic approaches to explore search space for an approximate solution in a computationally efficient manner [14]. However, designing and encoding effective heuristics into the search procedure is problem-dependent and largely relies on domain knowledge. Alternatively, metaheuristic approaches [15], [16] have been designed that provide high-level problem-independent search strategies to develop heuristic search algorithms rather than using hand-crafted heuristics.

Several recent studies have used machine learning techniques to learn heuristics or decision-making policies for task allocation and scheduling [17]–[20]. Particularly policy learning, a subfield of machine learning that aims to solve sequential decision-making problems, has been extensively exploited in dynamic task assignment problems. Gombolay et al. [18] proposed an apprenticeship scheduling method to learn domain expert heuristics for a class of scheduling problems. Ingimundardottir & Runarsson [19] used imitation learning to learn dispatching rules that are based on hand-crafted features

B. Adhikari and C. Jiang are with the Department of Electrical and Computer Engineering, University of Wyoming, Laramie, WY 82071 USA (e-mails: badhika1@uwyo.edu, cjiang1@uwyo.edu).

S. Ranashing, B. A. Miller and V. D. Novak are with the Department of Electrical Engineering and Computer Science, University of Cincinnati, Cincinnati, OH 45221 USA (e-mails: ranashsv@ucmail.uc.edu, mille7ba@ucmail.uc.edu, novakdn@ucmail.uc.edu).

This work is supported by the National Science Foundation under grant no. 2024813.

describing scheduling states. Zhang et al. [20] used a graph neural network (GNN) to model a job-shop scheduling problem and trained the scheduling policy via deep reinforcement learning.

In this paper, we investigate the problem of dynamic patient-robot task assignment and scheduling for a robotic rehabilitation gym with the goal of optimizing the group rehabilitation outcome. In our problem, a patient's skill acquisition is considered a stochastic process and the model describing the skill improvement over time is unknown. As a result, the objective function specifying the goal of task assignment and scheduling is not given explicitly in terms of assignment and scheduling decision variables and can only be evaluated numerically. We thus cast the task assignment and scheduling problem as a sequential decision-making problem and propose to exploit machine learning techniques to learn assignment and scheduling heuristics and policies from domain expert demonstrations. The problem considered in this paper differs from the class of scheduling problems studied in previous works [18]–[20] in that neither the precedence nor the duration of tasks performed on each agent is given a priori. Both precedence and duration should be determined by the policy to arrive at the optimal assignments and schedules.

The contributions of this paper are threefold. 1) We developed a sequential decision-making formulation for the dynamic patient-robot assignment problem in a robotic gym; 2) We proposed a two-stage assignment approach and used a neural network model that learns from expert demonstrations to predict pairwise priority; 3) We developed a robotic gym simulator to generate synthetic datasets in the absence of data from real robotic rehabilitation gyms.

The remainder of this paper is organized as follows. Section II presents the robotic gym scenario and the dynamic patient-robot assignment problem. Section III discusses our proposed approach. The results and discussion are presented in Section IV. We conclude our work in Section V.

II. Problem Definition

A. Scenario Description

For purposes of this study, a robotic gym consists of M patients and N robots working together for T time steps. Each patient has K skills to be improved corresponding to different functional abilities – e.g., hand function, shoulder function, etc. Before each time step, each patient is assigned to a specific robot and works with it for that time step. The following scenario constraints have been implemented regarding scheduling:

- Each session has a fixed duration of T time steps, with all patients beginning and ending the session simultaneously.
- Each robot only trains a single skill, and each skill is only trained by a single robot. Thus, $N = K$.
- At any given time step, only one patient can use a given robot, and a given patient can only use a single robot.
- All patients are assigned to robots before each time step simultaneously. Each patient must use the robot for the entire time step, with no switching possible mid-step.
- Once a patient-robot assignment is done, there are no further choices that need to be made for the robot – for example, it is not necessary to set the exercise difficulty level.

The patients' skill levels are not directly measurable but are reflected in the patient's performance during a time step. In the real world, performance could be measured using various metrics - task success, exercise intensity or motion quality [21], [22]. For purposes of this study, we assume that only task success rate is measured in each time step and is visible to the patient-robot assignment system after the time step ends. This success rate is imperfectly linked to the patient's skill level in the skill trained by the robot.

B. Dynamic Patient-Robot Assignment

For the scenario described in the previous subsection, we define a set of robots to be $R = \{r_1, r_2, \dots, r_N\}$, a group of patients $P = \{p_1, p_2, \dots, p_M\}$, and a set of motor skills $S = \{s_1, s_2, \dots, s_K\}$. For each patient $p_i \in P$, a feature set $G_{p_i}(t) = \{g_{p_i}^1(t), g_{p_i}^2(t), \dots, g_{p_i}^N(t)\}$, measured at each time step $t \in [1, T]$, is defined as the set of success rate $g_{p_i}^{r_j}(t)$ of the patient on each robot $r_j \in R$. The skill level of patient p_i on robot r_j at time step $t \in [1, T]$ is denoted by $S_{p_i}^{r_j}(t)$. Both success rate and skill level are evaluated on a continuous scale of [0,100]. The objective of the dynamic patient-robot assignment is to find a schedule of the patients' skill training on the robots that optimizes the overall group skill gain, defined by $\sum_{t=1}^{T} \sum_{p_i=1}^{M} \sum_{r_j=1}^{N} \Delta S_{p_i}^{r_j}(t)$, over the training period $[0, T]$. Here, $\Delta S_{p_i}^{r_j}(t) = S_{p_i}^{r_j}(t) - S_{p_i}^{r_j}(t-1)$ represents the skill gain obtained as the difference between the skill level at time t and skill level at previous time step $t - 1$. This problem can be framed as a sequential decision-making problem where the optimal schedule can be determined by solving the patient-robot assignment problem sequentially for every time step. The problem then amounts to finding a scheduling policy that at each time step automatically assigns robots to patients based on the patients' current features.

To address the defined dynamic patient-robot assignment problem, we propose to learn a scheduling policy that can reproduce the underlying strategies employed by a domain expert (e.g., occupational therapist) and achieve comparable patient training outcomes to those yielded by the domain expert's schedules. The scheduling policy is learned from a dataset of domain expert demonstrations via supervised learning. As real data from robotic rehabilitation gyms does not yet exist, we have also developed a simulator that generates a synthetic dataset as a basis for policy learning.

III. Approach

A. Dataset

1) Robotic Gym Simulator. A robotic gym simulator is created to simulate the dynamic process of the scenario described in Section II.A and generate synthetic data. The simulator takes patient-robot assignments given by a user (e.g., a domain expert) as input to update the skill level $S_{p_i}^{r_j}(t)$ and the success rate $g_{p_i}^{r_j}(t)$ at each time step t. Only success rate is returned and visible to the user for making patient-robot assignments. The initialization and updating of skill level and success rate are given as follows.

Skill level. The skill level is initialized at the start of the simulation using the following equation.

$$S_{p_i}^{r_j}(0) = L_{p_i} + d_{s_0}, \forall p_i \in P, r_j \in R \tag{1}$$

where $L_{pi} \in [0,100]$ is defined as the overall impairment level whose value is generated randomly between 16-40. The boundary values for impairment generation are kept lower to allow improvement during training; d_{s_0} is a random number sampled from a normal distribution with a mean of 5 and standard deviation of 3, i.e., $d_{s_0} \sim N(5, 3^2)$. The skill level is then updated in subsequent time steps using the following equation.

$$S_{p_i}^{r_j}(t) = S_{p_i}^{r_j}(t-1) + I_{p_i}^{r_j} + d_s, \forall p_i \in P, r_j \in R \tag{2}$$

where d_s is a random number sampled from a uniform distribution $U(-1,1)$ and $I_{p_i}^{r_j}$ is defined as the skill improvement which is given by

$$I_{p_i}^{r_j} = \begin{cases} 0.00102 G_{p_i}^{r_j}, \text{for } G_{p_i}^{r_j} < 70 \\ 5e^{0.005\left(G_{p_i}^{r_j} - 69\right)^2}, \text{for } G_{p_i}^{r_j} \geq 70 \end{cases}, \forall p_i \in P, r_j \in R \tag{3}$$

For our simplified scenario, the improvement increases gradually based on the success rate until the success rate reaches 70, which is often considered as an optimal target point that balances motivation and challenge 0. After this point, the improvement decreases. This is done to produce diminishing returns in skill level [23].

Success rate. Each patient's success rate is updated as a function of skill level. The equation to calculate it is given below.

$$G_{p_i}^{r_j}(t) = S_{p_i}^{r_j}(t) + d_g, \forall p_i \in P, r_i \in R \tag{4}$$

For our simplified scenario, the success rate is directly proportional to the skill level with a small random component, $d_g \sim U(-4, 4)$, added to it.

2) Data Generation. For the data generation, first the domain expert is prompted for the number of time steps needed. The simulator then displays randomly generated initial values of the patients' success rate of all the patients $p_i \in P$ in all the robots $r_j \in R$. Based on the displayed data, the domain expert then assigns the patients to the robots for the next time step. After the assignment is done, the values of the success rates are updated based on the assignment, and the new values for the subsequent time step are displayed to the domain expert. The domain expert repeats this process until the end of the desired number of time steps. At the end of the data generation, we obtain a data file with the feature set consisting of the success rate of all the patients as the input and the patient-robot assignment as the output for all time steps, with the assignment in each time step being independent of the other.

B. *Learning Scheduling Policy from Demonstrations*

In this section, we present a learning model that can learn from the synthetic data obtained as described in the previous subsection to make patient-robot assignments and produce comparable patient training results to those yielded by a domain expert. We propose a two-stage procedure to make assignments at each time step. In the first stage, a highest-priority patient for each robot is predicted using a neural network classifier trained by the demonstration dataset. Within the first stage, we adopt **two approaches** for the priority prediction: we train 1) multiple classifiers, one for each robot, and 2) a single classifier that will be used for all robots. In the second stage, if multiple robots have the same highest-priority patient, we resolve the conflict to determine which robot should be assigned to that patient.

1) Priority Prediction. We adopt the pairwise comparison method [18], which determines the priority between two patients by comparing their features. The advantage of the pairwise approach is that it helps to learn the rationales behind the assignment priority from the difference between the features of scheduled and unscheduled patients. The classifier for priority prediction, denoted as $f(p_i, p_x) \in \{0,1\}$, takes as input the features of two patients, p_i and p_x, and produces a binary label indicating whether patient p_i has a higher priority over patient p_x for a robot.

The data samples for training the pairwise classifier consist of both positive samples and negative samples. For positive samples, the input element, $\phi_{(p_i, p_x)}$, is the difference between the features of the assigned patient p_i and the unassigned patients p_x at each time step, and the corresponding output label, $y_{(p_i, p_x)}$, is set to 1. For negative samples, the input element, $\phi_{(p_x, p_i)}$, is the difference between the features of the unassigned patient p_x and the assigned patients p_i at each time step, and the corresponding output label, $y_{(p_x, p_i)}$, is set to 0. Since the classifier predicts priority for a single robot r_j at a time, only features associated with the robot r_j are used to create the input element. That is, for every robot $r_j \in R$ at time step t, we prepare the positive and negative samples as follows,

$$\phi_{(p_i, p_x)} = g_{p_i}^{r_j}(t) - g_{p_x}^{r_j}(t),\ y_{(p_i, p_x)} = 1,$$
$$\forall p_i \in P \text{ and } p_i \text{ assigned to robot } r_j,\ \forall p_x \in P \setminus p_i \tag{5}$$

$$\phi_{(p_x, p_i)} = g_{p_x}^{r_j}(t) - g_{p_i}^{r_j}(t), y_{(p_x, p_i)} = 0,$$
$$\forall p_i \in P \text{ and } p_i \text{ assigned to robot } r_j, \forall p_x \in P \setminus p_i \tag{6}$$

With the pairwise training samples created in the form of (5) and (6), we propose **two approaches** to predict the highest-priority patient for each robot at each time step. The first approach trains multiple classifiers, one for each robot in the scenario. That is, for an *N*-robot scenario, there are *N* separate classifiers to predict the highest-priority patient for each robot. The second approach trains a single classifier that is used for all the robots. The output of the classifier is the prediction of patient priority for the robot that the input features are associated with. Both approaches use the same neural network architecture discussed in subsection C.

While using the neural network classifier with each patient's pairwise training sample as described in subsection B.1), we obtain the output 1 if the patient is chosen and 0 if the patient is not chosen by the classifier. Each patient will obtain $M-1$ priority labels for a robot, one for each pairwise comparison of that patient and every other patient. Thus, for each robot r_j in a time step, we obtain the data, denoted by $count_{p_i}^{r_j}$, representing how many times a patient p_i has been chosen for robot r_j after being compared with every other patient. We also obtain the probability for each count produced by the sigmoid function in the final layer of the neural network. Then, we can take the mean of all probabilities of patient p_i to determine a mean probability, $Pr(p_i)$, of all the pairwise comparisons where the patient p_i has been chosen by the classifier. Based on $count_{p_i}^{r_j}$, we can have a rank of priority patients for each robot r_j and obtain the highest-priority patient depending on which patient has the highest count for each robot. For patients with the same count, we use the mean probability, $Pr(p_i)$, to sort the rank of the patients.

2) Conflict Resolving. In our scenario, it is assumed that a single patient can only use a single robot in each time step. The results of the first stage can be such that multiple robots might have the same suggested highest-priority patient. To resolve this conflict, we choose the robot r_j^* for which the patient has the highest mean probability value of all the counts. Subsequently, for other robots with the same priority patient, we look at the patients which have the next highest mean probability, and if they have not been selected as the priority patient for any other robot in the scenario, we choose that patient. The overall dynamic assignment and scheduling procedure is summarized in Algorithm 1.

Algorithm 1 Dynamic Assignment and Scheduling

```
for t = 0 to T
    // Stage 1 – Determine highest-priority patient
    for all robot r_j ∈ R do
        // Choose highest priority patient p_j based on
        // pairwise comparison
        p_i* ← argmax_{p_i ∈ P} Σ_{p_x ∈ P\p_i} f(p_i, p_x)
        Return count_{p_i*}^{r_j} and Pr(p_i*) associated with p_i*
    end for
    // Stage 2 – If multiple robots have same priority
    // patient
    for patient p_j shared by multiple robots {r_j'} ∈ R do
        // Choose the robot for which the patient has
        // highest mean probability
        r_j* ← argmax_{r_j ∈ {r_j'}} Pr(p_j)
        For the remaining robots with the same priority
        patient, choose the next patient with the highest
        mean probability not taken by another robot
    end for
end for
```

C. *Neural Network Classifier Architecture*

The neural network classifier consists of 1 input layer with hyperbolic tangent activation function, 1 output layer with sigmoid activation function, and 1 hidden layer with hyperbolic tangent activation function. The number of neurons in the input layer is equal to 1. The number of neurons in the output layer is equal to 1. The hidden layer consists of 32 neurons. The sigmoid function in the final layer of the neural network classifier provides the probability of the output label. An Adam optimizer is used with a learning rate of $\eta = 0.01$ and a binary cross entropy loss function to train the model.

D. *Evaluation Methodology*

Both proposed approaches were evaluated on a synthetic dataset. The dataset consisted of training sessions with 5 patients and 5 robots. Five different skills were considered, with each skill trained on a different robot. Ten data files, each consisting of a 12-time step session with different initial patient skills, were created. For the synthetic dataset, co-author VDN (a rehabilitation engineering researcher) worked with the simulator as the "domain expert". She was able to see the equations of the model before working with the simulator, and (self-reported) aimed to target each patient's lowest skill at each time step. Before generating the 10 data files used for the study, she generated 10 "practice" files that were not used for the study.

The files were split into training and test using a leave-one-out cross-validation method and the results were obtained for all 10 sessions. Two evaluation metrics were used. First, we compared the patient-robot assignment made by the domain expert in each time step with the assignments made by our two approaches for every session. By comparing each individual patient-robot assignment in every time step, we obtained the percentage accuracy between the domain expert scheduling assignment and the scheduling assignment of our approaches. Second, we took the overall skill growth of each patient obtained during a session and took a mean of those values to get the mean skill growth obtained during a session for the patient-robot assignment made by our approaches, domain expert, and a random assignment. To obtain all four results, all the patients were initialized with the same skill level for each session.

IV. Results

The results of 10 simulated sessions are shown in Table I and Table II. Table I shows the classification accuracy of patient-robot assignments done by the two approaches and assignments done by the domain expert. A paired t-test on the data from Table I found that Approach 1 yielded significantly higher accuracy than Approach 2 ($p < 0.001$).

Table II shows the mean skill growth averaged over all the patients in a session for 10 different sessions for different approaches. A one-way repeated-measures analysis of variance on the data from Table II found that the approaches were significantly different ($p < 0.001$) and that the random assignment was significantly worse than all three other approaches ($p < 0.001$ in all three post-hoc Holm-Sidak tests).

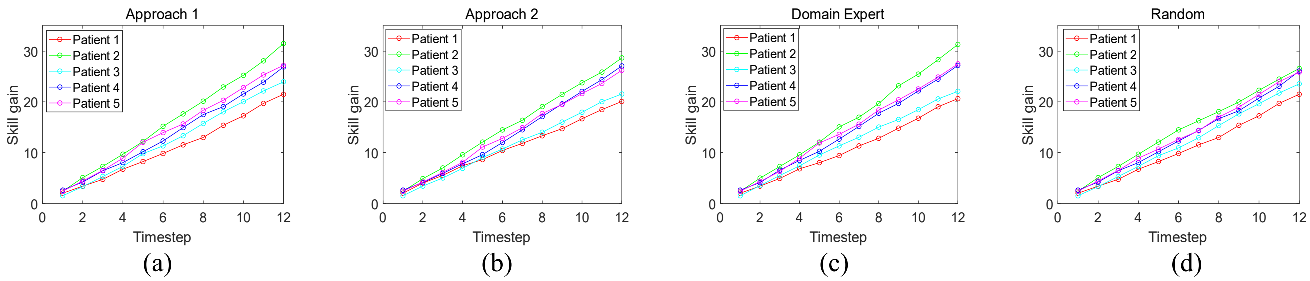

Figure 1. Total Skill gain for each patient over one session for patient-robot assignments by (a) Approach 1 (multiple classifiers), (b) Approach 2 (single classifier), (c) domain expert, and (d) random assignment.

TABLE I: ACCURACY RESULTS OF PATIENT-ROBOT ASSIGNMENT BETWEEN DOMAIN EXPERT AND THE TWO APPROACHES: APPROACH 1 (MULTIPLE CLASSIFIERS) AND APPROACH 2 (SINGLE CLASSIFIER)

Session	Approach 1	Approach 2
1	87.1%	82.1%
2	89.2%	80.2%
3	84.6%	74.6%
4	83.9%	77.8%
5	85.3%	78.4%
6	90.3%	79.5%
7	88.5%	78.3%
8	87.6%	81.1%
9	89.4%	76.5%
10	87.9%	78.3%
Mean ± SD	87.4 ± 2.4%	78.7 ± 2.3%

TABLE II. MEAN SKILL GROWTH OF THE PATIENTS OVER 12 TIME STEPS FOR PATIENT-ROBOT ASSIGNMENTS DONE BY APPROACH 1 (MULTIPLE CLASSIFIERS), APPROACH 2(SINGLE CLASSIFIER), DOMAIN EXPERT AND RANDOM ASSIGNMENT

Session	Mean Skill Growth			
	Approach 1	***Approach 2***	***Domain expert***	***Random***
1	**33.4**	31.5	32.8	27.1
2	**27.8**	26.6	27.2	25.2
3	25.1	24.6	**25.3**	22.8
4	30.2	**30.5**	29.8	27.1
5	29.3	28.6	**29.6**	25.2
6	**30.9**	29.4	29.7	27.7
7	28.1	27.6	**28.4**	25.1
8	**32.8**	32.6	31.7	27.1
9	23.8	23.5	**24.8**	21.7
10	**26.6**	24.7	25.5	22.9
Mean ± SD	**28.8 ± 3.1**	27.9 ± 3.1	28.5 ± 2.8	25.2 ± 2.1

Fig. 1 shows the total skill gain for each patient over a session for patient-robot assignments done by the two presented approaches, the domain expert, and a random assignment. Fig. 2 shows the box plot of mean skill growth of all patients for all 10 sessions.

V. DISCUSSION

From Table I, we can see that Approach 1 has a mean accuracy of 87.4% and Approach 2 has a mean accuracy of 78.7% when compared with the patient-robot assignments done by the domain expert. This indicates that both approaches can learn the scheduling policy from domain expert demonstration. Additionally, Approach 1 consistently performs better than Approach 2. The difference in performance is likely due to the use of multiple models in Approach 1. In Approach 1, each model was trained with a robot-specific feature set and could find the feature-priority mapping that achieves good prediction accuracy for that individual robot. In contrast, the single model in Approach 2 was trained so that it can generalize to different robots. However, the generalizability was obtained at the cost of lower prediction accuracy for individual robots.

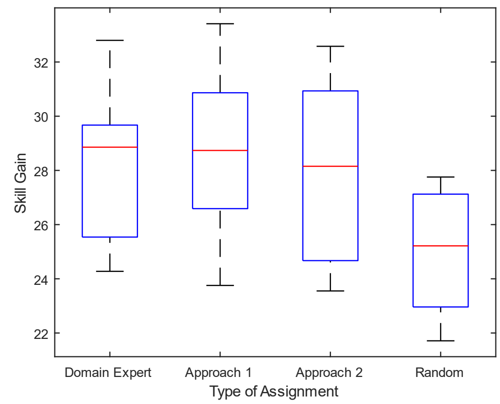

Figure 2. Box plot of mean skill gain for sessions 1-10 for domain expert, Approach 1 (multiple classifiers), Approach 2 (single classifier) and random assignment.

Table II and Fig. 2 show the data about mean skill growth of all the patients for 10 different sessions. Both approaches give comparable results to that of the domain expert while the results of the random assignment were worse than all the other approaches. This is further evidenced in Fig. 1 where the overall skill gains in both approaches are comparable to that of the domain expert assignment while the random assignment has slightly worse performance than the others. Since the random numbers to update the skill levels and success rates are generated dynamically, our approaches may over/underperform the domain expert due to the randomness. It's worth mentioning that the patients may not benefit equally by maximizing the group skill gain. This can be addressed by considering multiple optimization objectives to balance the skill improvement across all patients.

It is worth noting that the underlying model is relatively simple: there is only one measurable variable (i.e., success rate) that is linked to skill level with a fairly straightforward relationship. In practice, estimating patient skill level is a significant challenge and may involve interplay between multiple patient characteristics (e.g., injury type), robot settings (e.g., assistive strategy, task difficulty [21], [24]), measurable quantities (e.g., success, workload, motion quality [21], [22]) and unmeasurable quantities (e.g., patient motivation [24]). Furthermore, skill improvement depends only on success rate in our simplified model, but would depend on all the above characteristics in the real world as well. In the next stage of the work, we will thus aim to test the approaches on a synthetic dataset with a more complex underlying model.

Finally, only one domain expert (a co-author of the work) created the dataset used for evaluation. Different experts may have different scheduling policies that may be easier or harder to learn as well as more or less effective than the current one. Thus, in future work, we will investigate our approaches with multiple domain experts.

VI. Conclusion

We proposed a model that can learn dynamic patient-robot task assignment and scheduling from a domain expert using pairwise training samples and neural network. Two different approaches were used to get the patient priority. One approach used N different neural network models for N robots, whereas the other approach used a single model to determine the patient priority. We validated our approaches using leave-one-out cross-validation and collected the statistics on mean skill growth and percentage accuracy with the domain expert demonstration. The dataset for the model was created using a robotic gym simulator that emulated a rehab gym scheduling scenario. The results of both approaches showed that they were able to create a patient-robot assignment that produces results comparable to that of the domain expert.

Our approaches provide a way to learn policies that enable automated patient-robot assignment and scheduling for a rehabilitation gym. The approaches learn policies in a supervised manner and are applicable when the heuristic rules are hard to encoded but can be easily demonstrated by domain experts. Our future work will explore unsupervised learning paradigms to learn optimal policies that could yield better rehabilitation training outcomes than those obtained by the heuristic rules from domain experts.

References

[1] D. Novak, A. Nagle, U. Keller, and R. Riener, “Increasing motivation in robot-aided arm rehabilitation with competitive and cooperative gameplay,” *J. Neuroeng. Rehabil.*, vol. 11, p. 64, 2014.

[2] K. Baur, A. Schättin, E. D. de Bruin, R. Riener, J. E. Duarte, and P. Wolf, “Trends in robot-assisted and virtual reality-assisted neuromuscular therapy: a systematic review of health-related multiplayer games,” *J. Neuroeng. Rehabil.*, vol. 15, no. 107, 2018.

[3] E. B. Küçüktabak, S. J. Kim, Y. Wen, K. Lynch, and J. L. Pons, “Human-machine-human interaction in motor control and rehabilitation: a review,” *J. Neuroeng. Rehabil.*, vol. 18, p. 183, 2021.

[4] K. Bustamante Valles, S. Montes, M. de J. Madrigal, A. Burciaga, M. E. Martínez, and M. J. Johnson, “Technology-assisted stroke rehabilitation in Mexico: a pilot randomized trial comparing traditional therapy to circuit training in a robot/technology-assisted therapy gym,” *J. Neuroeng. Rehabil.*, vol. 13, p. 83, 2016.

[5] I. Aprile *et al.*, “Upper limb robotic rehabilitation after stroke: a multicenter, randomized clinical trial,” *J. Neurol. Phys. Ther.*, vol. 44, no. 1, pp. 3–14, 2020.

[6] I. Aprile, C. Pecchioli, S. Loreti, A. Cruciani, L. Padua, and M. Germanotta, “Improving the efficiency of robot-mediated rehabilitation by using a new organizational model: an observational feasibility study in an Italian rehabilitation center,” *Appl. Sci.*, vol. 9, no. 24, p. 5357, 2019.

[7] G.A. Korsah, A. Stentz, and, M.B. Dias, “A comprehensive taxonomy for multi-robot task allocation,” *Int. J. Rob. Res.*, vol. 32, no. 12, pp.1495–1512, 2013.

[8] A. Casalino, A. M. Zanchettin, L. Piroddi, and P. Rocco, “Optimal scheduling of human-robotcollaborative assembly operations with time Petri nets,” *IEEE Trans. Autom. Sci. Eng.*, vol. 18, no. 1, pp. 70–84, 2021.

[9] K. E. C. Booth, G. Nejat, and J. C. Beck, “A constraint programming approach to multi-robot taskallocation and scheduling in retirement homes,” in *International Conference on Principles andPractice of Constraint Programming*, pp. 539–555, 2016.

[10] M. Gombolayet al., “Robotic assistance in the coordination of patient care,” *Int. J. Rob. Res.*, vol. 37, no. 10, pp. 1300–1316, 2018.

[11] M. Gombolayet al., “Human-machine collaborative optimization via apprenticeship scheduling,”*J.Artif. Intell. Res.*, vol. 63, pp. 1–49, 2018.

[12] W.-Y. Ku and J. C. Beck, “Mixed Integer Programming models for job shop scheduling: Acomputational analysis,” *Comput. Oper. Res.*, vol. 73, pp. 165–173, 2016.

[13] F. Rossi, P. van Beek, and T. Walsh. *Handbook of Constraint Programming (Foundations of Artificial Intelligence)*. Elsevier Science Inc., New York, NY, USA, 2006.

[14] E. Castro and S. Petrovic, “Combined mathematical programming and heuristics for aradiotherapy pre-treatment scheduling problem,” *J. Sched.*, vol. 15, no. 3, pp. 333–346, 2012.

[15] A. Kushleyev, D. Mellinger, C. Powers, and V. Kumar, “Towards a swarm of agile microquadrotors,” *Auton. Robots*, vol. 35, no. 4, pp. 287–300, 2013.

[16] L. Zhang and T. N. Wong, “An object-coding genetic algorithm for integrated process planning andscheduling,” *Eur. J. Oper. Res.*, vol. 244, no. 2, pp. 434–444, 2015.

[17] I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio, “Neural combinatorial optimization with reinforcement learning,” in *5th International Conference on Learning Representations(ICLR)*, 2017.

[18] M. Gombolay, R. Jensen, J. Stigile, S. H. Son, and J. Shah, “Apprenticeship scheduling: Learningto schedulefrom human experts,” in *AAAI Press/International Joint Conferences on Artificial Intelligence*, 2016.

[19] H. Ingimundardottir and T. P. Runarsson, “Discovering dispatching rules from data using imitation learning: A case study for the job-shop problem,” *J. Scheduling*, vol. 21, no. 4, pp. 413–428, Aug. 2018.

[20] C. Zhang, W. Song, Z. Cao, J. Zhang, P. Siew Tan, and X. Chi, “Learning to dispatch for job shop scheduling via deep reinforcement learning,” In *Advances in Neural Information Processing Systems*, vol. 33, pp. 1621–1632, 2020.

[21] M. S. Cameirão, S. B, Badia, E. D. Oller and P. F. Verschure, “Neurorehabilitation using the virtual reality based Rehabilitation Gaming System: methodology, design, psychometrics, usability and validation,” *J. Neuroeng. Rehabil.,* vol. 7, no. 1, pp. 1-14, 2010.

[22] S. Balasubramanian, A. Melendez-Calderon, A. Roby-Brami and E. Brudet, “On the analysis of movement smoothness,” *J. Neuroeng. Rehabil.,* vol 12, no. 1, pp. 1-11, 2015.

[23] K.M. Newell, Y.T. Liu, and G Mayer-Kress, “Time scales in motor learning and development,” *Psychol. Rev.,* vol. 108, no. 1, pp. 57-82, 2001.

[24] R. Colombo, F. Pisano, A. Mazzone, C. Delconte, S. Micera, M. C. Carrozza, P. Dario and G. Minuco, “Design strategies to improve patient motivation during robot-aided rehabilitation,” *J. Neuroeng. Rehabil.,* vol. 4, no. 1, pp. 1-12, 2007 .

[25] J-C. Metzger, O. Lambercy, A. Califfi, D. Dinacci, C. Petrillo, P. Rossi, F. M. Conti and R. Gassert, “Assessment-driven selection and adaptation of exercise difficulty in robot-assisted therapy: a pilot study with a hand rehabilitation robot,” *J. Neuroeng. Rehabil.,* vol. 11, no 154, 2014.