Towards On-Demand Virtual Physical Therapist: Machine Learning-Based Patient Action Understanding, Assessment and Task Recommendation

Wenchuan Wei, *Student member, IEEE*, Carter McElroy, and Sujit Dey, *Fellow, IEEE*

***Abstract*—In this paper, we propose a machine learning-based virtual physical therapist (PT) system to enable personalized remote training for patients with Parkinson's disease (PD). Three physical therapy tasks with multiple difficulty levels are selected to help patients with PD improve balance and mobility. Patients' movements are captured by a Kinect sensor. Criteria for each task are carefully designed by our PT co-author such that the patient's performance can be evaluated in an automated manner. Given the patient's motion data, we propose a two-phase human action understanding algorithm TPHAU to understand the patient's movements, and an error identification model to identify the patient's movement errors. To enable automated task recommendation, a machine learning-based model is trained from real patient and PT data to provide accurate, personalized, and timely task update recommendation for patients with PD, thereby emulating a real PT's behavior. Real patient data have been collected in the clinic to train the models. Experiments show that the proposed methods achieve high accuracy in patient action understanding, error identification and task recommendation. The proposed virtual PT system has the potential of enabling on-demand virtual care and significantly reducing cost for both patients and care providers.**

***Index Terms*—Action Understanding, Machine Learning, Parkinson's Disease, Physical Therapy, Recommendation System**

I. Introduction

Parkinson's disease (PD) is the most common movement disorder. It affects about 1 million people in the US and 10 million worldwide [1]. The combined direct and indirect cost of PD is estimated to be nearly $25 billion per year in the US alone [1]. Physical therapy is an essential treatment for patients with PD. Traditional physical therapy requires regular visits to the physical therapist (PT) (shown in Fig. 1), which may be expensive and inconvenient for patients with PD due to factors such as insufficient insurance coverage, impaired mobility, etc. For traditional physical therapy, a PT selects the training tasks, instructs the patient on how to perform the tasks, identifies and corrects the patient's errors, and regularly updates the tasks, all in the clinic. After the PT session, the patient is expected to practice the training tasks at home by following written instructions provided by the PT. However, the patient's performance and adherence to the tasks cannot be tracked at home without the supervision of the PT. Practicing a task with incorrect technique is not only ineffective for motor learning, it may also cause injury due to the impaired mobility of patients with PD. King et al. has shown poor outcomes with unsupervised home-based exercise programs for patients with PD [2]. Furthermore, the training tasks cannot be updated until the patient's next PT visit. Continuing to practice the same training task, which may not be suitable any more for the current state of the patient, could limit the patient's progress or even reinforce motor learning in a negative way. To address these problems, several automated training systems have been developed to motivate patients and monitor their movements at home using motion capture sensors [3-6, 13]. However, these systems are not aimed at performance accuracy, and cannot provide personalized task recommendation for patients.

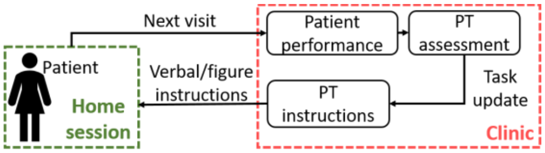

Fig. 1. Traditional physical therapy treatment procedure.

In this paper, we propose an on-demand virtual PT system, shown in Fig. 2. The patient can use the cloud-based system we proposed in [7], where a Kinect sensor [8] was used to capture the patient's movements and avatars are created to provide instructions and feedback. Instead of aligning the patient's motion data with PT templates to evaluate the patient performance like [7], we propose a two-phase human action understanding (TPHAU) algorithm that can understand the patient's sub-actions in performing the task and a Support Vector Model (SVM) based method to identify the patient's errors. Moreover, based on the patient's error and some subjective factors (e.g., age, discussed in Section III-D-1), a machine learning-based task recommendation model is proposed to provide automated task update recommendation for patients. Based on the recommendation results, either a new task or a guidance video will be rendered on the cloud and sent to the patient's device. The PT can remotely supervise the entire process. The proposed virtual PT system has the advantages of providing accurate, on-demand and personalized care. It has the potential of significantly reducing clinic visit requirements while offering continuous care, thereby reducing cost and expanding care for economically disadvantaged and rural patient populations. To validate the effectiveness of the proposed methods, we have collected real patient data in the Neurological Rehabilitation Clinic, UC San Diego Health. The proposed models are trained from the collected data and experimental results show that the proposed methods achieve high accuracy in patient action understanding, error identification and task recommendation.

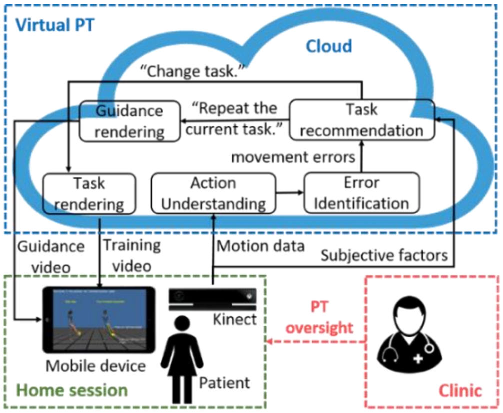

Fig. 2. The proposed on-demand virtual PT system.

A preliminary version of this work has been reported in [11], which introduced the PT-defined training tasks and criteria for patients with PD, and proposed the action understanding and error identification methods. In [11], any task update would still need to be performed manually by the PT at the clinic. In comparison, this paper enhances [11] to propose a machine learning-based task recommendation model to enable on-demand and personalized task recommendation for patients with PD. The task recommendations can be fully automated, or if desired, the system may require remote supervision and approval by the PT.

The remainder of this paper is organized as follows: Section II reviews related work. Section III introduces the methods proposed in the virtual PT system. Section IV presents the experimental results. Section V concludes the paper and discusses future work.

II. Related Work

A. Automated training systems for patients with PD

With the development of motion capture sensors, more and more sensor-based automated training systems have been developed to improve the effectiveness of home training for patients with balance and mobility problems. Hssayeni et al. [3] used wearable sensors to identify motor fluctuations in patients with PD during a variety of daily living activities. Stack et al. [4] used wearable sensors to detect subtle instability in patients with PD. However, wearable sensors attached on the body may cause extra burden to patients with PD due to their impaired mobility. Therefore, camera-based sensors were considered more convenient in monitoring the movements of patients with PD. Galna et al. [12] proved the high accuracy of the Kinect sensor in measuring clinically relevant movements in patients with PD. Galna et al. [5] and Pompeu et al. [6] designed two game-based training systems using Kinect and proved their feasibility and safety for patients with PD. However, the game-based training systems are designed to motivate the patients and cannot enable careful monitoring of desired patient performance and subsequent task recommendation like a PT does. Lin et al. [13] developed a Kinect-based rehabilitation system to assist patients with movement disorders and balance problems. However, the performance evaluation method proposed in [13] failed to consider the patient's reaction delay as it simply compared the patient's movements with the standard movement frame by frame. Most of these training systems provide uniform training for patients and cannot provide accurate evaluation, personalized feedback, and most importantly, task recommendation based on the patient's performance like a PT at the clinic. In comparison, our proposed virtual PT system provides accurate movement understanding, error identification, and personalized task recommendation, rendering our system unique. In addition, the cloud-based system can be used at any place at any time to enable on-demand virtual care, with the potential to enable personalized physical therapy with better effectiveness and compliance, while lowering cost and increasing patient participation.

B. Human action understanding

To enable automated performance evaluation, the first step is to understand the patient's movements/actions. Generally, human action understanding includes two categories: 1) Action recognition, which is the classification of an action from videos [14, 15]. However, recognizing what task the patient is performing is insufficient. We need to understand the movement details and identify the patient's movement errors. 2) Action detection/segmentation, which refers to locating actions of interest in space and/or in time [16, 17]. Most studies in this area focus on the detection of long action segments. In [16] and [17], a detected segment is considered correct if the overlap between it and the ground-truth action segment is over 40%, as this threshold is consistent with visual inspection. However, the sub-actions discussed in this paper (see Section III-A-2) are much shorter in time length and closer to each other (i.e., the pause between adjacent sub-actions is negligible), which makes the segmentation much more challenging. In this paper, we propose the TPHAU algorithm to accurately detect/segment the patient's sub-actions, which will be discussed in Section III-B.

C. Automated Recommendation systems

With the rapid development of artificial intelligence, more and more automated recommendation systems have been developed to enable optimized and personalized user experiences, e.g., friend recommendation in social networks [9, 29], ad recommendation [10, 30], etc. However, little research has been conducted to develop automated task recommendation systems for healthcare applications. To the best of our knowledge, we are the first to achieve automated task recommendation for patients with PD. The proposed virtual PT system is trained from real patient and PT data, thus it enables accurate and personalized task recommendation for patients with PD.

III. Methods

A. Kinect-based Automated Training System for Patients with Parkinson's Disease

In this section, we first introduce the training tasks selected by our PT co-author for patients with PD, then discuss how the proposed training system can identify the patient's movement

TABLE I. TASKS AND DIFFICULTY LEVELS. FROM LEFT TO RIGHT: SQUAT (SQ), FORWARD LUNGE (FL), BACKWARD LUNGE (BL).

Levels of SQ	**Hand support**	**Squatting angle**	**Levels of FL**	**Hand support**	**Length of step**	**Levels of BL**	**Hand support**	**Length of step**
SQ1	Yes	Small	FL1	Yes	Small	BL1	Yes	Small
SQ2		Large	FL2		Large	BL2	Step back with hand support, then take hands off	Large
SQ3	No	Small	FL3	No	Small	BL3	No	Small
SQ4		Large	FL4	Arms up	Large	BL4		Large

errors automatically. To avoid confusion, we would like to clarify the definitions of four terms: task, movement/action, repetition, and sub-action. Task is an exercise designed by the PT to train patients. Movement/action is the execution of the task by a patient, which may contain one or multiple repetitions. Each repetition can be further divided into several sub-actions, which will be introduced in Section III-A-2.

1) Tasks and Difficulty Levels

Based on the work of King et al. [18] describing sensorimotor agility training for patients with PD, our PT co-author has selected three balance/agility based tasks: squat (SQ), forward lunge (FL) and backward lunge (BL). For each task, four difficulty levels (level 1 ~ 4) are designed (see Table I). During a traditional PT session, a patient performs a given training task at a certain difficulty level . The PT inspects the patient's performance and decides if changes to the difficulty level is needed. For example, a patient who currently performs a squat exercise may progress to a more difficult variation of the squat if the initial difficulty level becomes too easy as the patient improves. The PT's assessments are based on self-designed criteria for each task. Criteria are based on different sub-actions of a given exercise movement, which will be introduced in Section III-A-2.

2) Sub-actions and Criteria

For each physical therapy task, the patient's movements can be divided into several sub-actions. For example, movements in FL include: 1) stand, 2) step forward, 3) maintain balance control, 4) return to the original position, 5) stand. Therefore, we define five sub-actions $S_1 \sim S_5$ in Table II, which apply to all the three tasks considered for patients with PD: SQ, FL and BL.

TABLE II. SUB-ACTIONS IN PATIENT'S MOVEMENTS

Sub-action	**Patient's movements**
S_1	Standing
S_2	Movement initiation: try to reach the target position
S_3	Balance hold: maintain balance control
S_4	Return to the original position
S_5	Standing

To evaluate the patient's performance in an automated and quantified way, we have defined some criteria for each task (i.e., the rules for evaluating the patient's performance). These criteria have been selected based on the expert PT's knowledge of compensatory movement strategies of patients with PD. For example, a common compensatory strategy that a patient with PD may use in FL is to bend the knee of the back leg, due to both strength and balance impairments. Therefore, the PT has defined "keep the back knee straight" as one of the criteria for FL. A task criterion is applicable to one or more sub-actions of the task. Table III shows the criteria defined by our PT co-author and the applied sub-actions for SQ, FL, and BL.

In the Kinect-based training system, the Kinect sensor captures 25 joints of the human skeleton with 3-D coordinates for each joint [8]. To enable automated action understanding and error identification, we first need to translate PT's criteria into some Kinect-captured quantities (KCQs). KCQs are quantities that can be derived from the joint coordinates captured by Kinect. In this paper, we define the following six KCQs for the three tasks. (Considering the difference in body size, we use normalized quantities, e.g., angles and normalized length of step.)

Thigh Angle (ThA): the angle between the thigh and the vertical direction. In SQ, we use the average of the left and right thigh angles to represent the squatting angle.

Trunk Angle (TrA): the angle between the trunk and the vertical direction. It represents the forward-leaning angle in SQ and can be used to check whether posture is tall in FL.

Trunk-Leg Angle (TrLA): the angle between the trunk and the back leg. In BL the patient should lean slightly forward thus keeping the trunk parallel with the back leg.

Knee Angle (KA): the angle between the thigh and the shank, representing whether the knee is straight.

Normalized Length of Step (NLoS): the distance between the two feet, normalized by the length of the leg.

Shank Angle (SA): the angle between the shank and the vertical direction, representing whether the shank is vertical.

Fig. 3 shows these KCQs. KCQs used in multiple tasks are shown in only one task for simplicity. The target value of each KCQ shown in Table III is either defined by the PT (e.g., *KA*: 180°) or derived from the PT's demonstration (e.g., *ThA*: 49° for small angle and 67° for large angle).

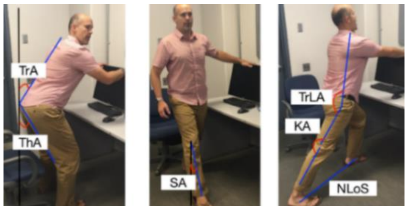

Fig. 3. Tasks and Kinect-captured quantities (KCQs). From left to right: Squat (SQ), Forward Lunge (FL), Backward Lunge (BL).

Given the KCQs, the patient's performance can be evaluated automatically by checking the KCQs in the applied sub-actions. In Section III-B, we will introduce how to segment the sub-actions in patient's movements.

B. Patient Action Understanding

Action understanding in the proposed system includes two steps: 1) Repetition detection. The patient may perform multiple

TABLE III. PT-Defined Criteria, Kinect-captured quantities (KCQs) and Applied Sub-actions forSquat (SQ), Forward Lunge (FL), Backward Lunge (BL).

SQ: PT's Criterion	KCQ	Applied sub-actions
Sit hips back towards a chair	*ThA*: 49° (small), 67° (large)	S_3
Lean forward	*TrA*: 22° (small), 27° (large)	S_3

FL: PT's Criterion	KCQ	Applied sub-actions
Keep the back knee straight	*KA* (back leg): 180°	S_2, S_3
Keep the posture tall	*TrA*: 0°	S_2, S_3, S_4
Length of step	*NLoS*: 0.47 (small), 0.79 (large)	S_3
Keep the front shank vertical	*SA* (front leg): 0°	S_3

BL: PT's Criterion	KCQ	Applied sub-actions
Keep the back knee straight	*KA* (back leg): 180°	S_3
Keep the trunk parallel with the back leg	*TrLA*: 0°	S_2, S_3, S_4
Length of step	*NLoS*: 0.48 (small), 0.78 (large)	S_3
Keep the front shank vertical	*SA* (front leg): 0°	S_2, S_3

repetitions on a task each time, thus we need to detect the starting point and endpoint of each repetition. 2) Sub-action segmentation, i.e., to segment the sub-actions in each repetition. To achieve this, we propose two Hidden Markov Models (HMMs) [19]: HMM-S for single repetition and HMM-M for multiple repetitions in Fig. 4 and Fig. 5. Details about the components of HMM are discussed in our preliminary work [11]. HMM-S consists of five hidden states S_1 to S_5. (Note that one state in the HMM model represents a sub-action in patient's movements, thus we use the same symbol S_i for both.) The state transfers from S_1 to S_5 and ends in S_5. For multiple repetitions, the state will transfer back to S_1 after S_4 and start a new repetition. Therefore, S_1 to S_5 are combined into one state in HMM-M. a_{ij} is the state transition probability, i.e., the probability of transferring from S_i to S_j.

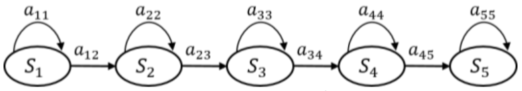

Fig. 4. HMM-S: the HMM model for single repetition.

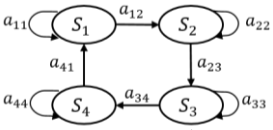

Fig. 5. HMM-M: the HMM model for multiple repetitions.

A key issue to be addressed for the HMM model is the HMM feature to be selected for the model. The HMM feature is the quantity we observe and use to infer the hidden states. It can be any subset of the joint coordinates, or quantities derived from the joint coordinates (like the six KCQs defined in Section III-A-2). For the two HMM models defined in this paper, the displacement d and velocity v of the primary moving body parts are selected as the HMM feature. In the task SQ, the patient bends his/her legs to move the hips up and down, thus *ThA* represents the movement and is used as the displacement d. In the tasks FL/BL, the patient moves one foot back and forth so *NLoS* can be used as the displacement d. The velocity v is calculated from d. Reasons for using the combination of d and v instead of any single variable as the HMM feature are discussed in [11].

Parameters of an HMM model include the state transition probability a_{ij}, emission probability $b_j(X)$ (i.e., the probability of observing X under state S_j), and the initial state distribution π_i (i.e., the probability that the Markov chain starts from state S_i). For HMM-S and HMM-M, parameters are estimated using supervised learning. Training data are collected from real patients with PD. For each training sample, five sub-actions in the movements (see Table II) are manually segmented. (Note that for HMM-M, S_1 includes the manually-labelled S_1 and S_5.) The transition probability a_{ij} is calculated as

$$a_{ij} = \frac{\text{number of transitions from } S_i \text{ to } S_j}{\text{number of transitions from } S_i},\ 1 \le i, j \le N. \qquad (1)$$

For the emission probability, we use the Gaussian Mixture Model (GMM) as

$$b_j(X) = \sum_{c=1}^{C} w_{jc} \mathcal{N}(\mu_{jc}, \Sigma_{jc}), \qquad (2)$$

where C is the number of mixture components, w_{jc}, μ_{jc}, Σ_{jc} are the weight, mean, and covariance of the *c-th* Gaussian component. Parameters of GMM are estimated from the training data using the Expectation-Maximization (EM) algorithm [22]. The GMM model of each sub-action/state is trained separately using the motion data in that state.

Given the model parameters $\lambda = \{a_{ij}, b_j(X), \pi_i\}$, our goal is to infer the hidden state sequence Q from any new observation sequence O. The Viterbi algorithm [23] is a dynamic programming algorithm for finding the most likely hidden state sequence Q^* of the observation O using

$$Q^* = \arg\max_{Q} P(Q \mid O, \lambda) = \arg\max_{Q} P(Q, O \mid \lambda). \qquad (3)$$

Based on the Viterbi algorithm, we propose a two-phase human action understanding (TPHAU) algorithm to detect the patient's repetitions and segment sub-actions in each repetition. In the first phase, the HMM-M model is used to detect the starting point and endpoint of each repetition. Considering the difference of displacement amplitude in different patients and in different repetitions, we apply repetition-based normalization on the displacement data d of the training samples (i.e., d in each repetition are normalized into [0, 1]). For the test samples, global normalization (i.e., d of the entire performance including multiple repetitions are normalized into [0, 1]) is used since the time interval for each repetition is unknown in the first phase. Then based on the trained HMM-M model, the hidden states of the test samples can be estimated by applying the Viterbi algorithm [23] and the patient's repetitions can be further inferred. Since S_1 is the boundary between two repetitions, the starting point of each repetition (except the first one) can be estimated as the midpoint

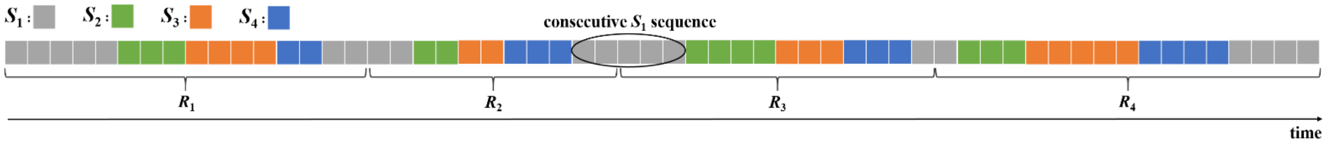

Fig. 6. Hidden state sequence obtained from the Viterbi algorithm [23]. Four repetitions R_1, R_2, R_3, R_4 are inferred.

of each consecutive S_1 sequence. Fig. 6 shows an example. Four repetitions R_1 ~ R_4 are detected from the hidden state sequence.

However, noise in the motion data may cause the detection of extra repetitions. There are two types of extra repetitions: 1) noise being detected as complete repetitions, 2) recognizing one repetition as two or more. Detailed discussion about extra repetitions can be found in our preliminary work [11]. To remove the extra repetitions, we analyze the Time Length (TL), the Amplitude of Displacement (AoD) (i.e., maximum of d), and the Displacement of Endpoint (DoE) (i.e., d at the endpoint of each repetition) of all the repetitions in the training data. The mean value μ_{TL}, μ_{AoD}, μ_{DoE} and standard deviation σ_{TL}, σ_{AoD}, σ_{DoE} are calculated. According to the three-sigma rule [24], a detected repetition is an outlier and considered as extra repetition if

$$\left|TL-\mu_{TL}\right|>3\sigma_{TL} \text{ or } \left|AoD-\mu_{AoD}\right|>3\sigma_{AoD} \text{ or } \left|DoE-\mu_{DoE}\right|>3\sigma_{DoE}. \tag{4}$$

An extra repetition is eliminated by merging into its previous or next repetition, whichever is closer to it (i.e., the one with fewer frames of S_1 between them). After removing extra repetitions, we use a second phase to segment sub-actions in each repetition. Although the state sequence obtained from the first phase also includes information about sub-actions in each repetition, the sub-action information is not accurate for the following reasons. In the first phase, global normalization is used thus the range of d in some repetitions may be smaller than [0, 1]. Different normalization methods on the training and test data will cause inaccuracy in state/sub-action segmentation. For example, in training data, d will always reach 1 in S_3 because of the repetition-based normalization. For a test sample where $d < 1$ in S_3, some frames at the beginning of S_3 may be detected as S_2. To solve this problem, we propose using a second phase to enhance the accuracy in sub-action segmentation. First, we normalize the displacement data d of each repetition that is detected in the first phase. Second, the HMM-S model is applied on each repetition. Since the HMM-S model is a left-to-right model for single repetition, it is guaranteed that no extra repetitions will be detected. Therefore, sub-actions can be segmented based on the hidden state results in the second phase. Fig. 7 shows the pseudo-code for the proposed TPHAU algorithm.

C. Movement Error Identification

In this section, we will introduce how to identify the patient's movement errors. For any task, the criteria used for evaluating the patient's performance have been defined by our PT co-author (see Table III). Criteria are independent of each other (i.e., whether the patient is performing correctly on one criterion is independent of his/her performance on the other criterion). Based on the repetition detection and sub-action segmentation results, the patient's movement errors can be identified by checking the value of his/her corresponding KCQs in the applied sub-actions of each criterion. For example, the criterion "keep the back knee straight" of FL applies to S_2 and S_3 (see Table III), so we just need to check the knee angle (KA) of the back leg for frames in S_2 and S_3. The patient's error in one frame e_{frame} is calculated as the difference between the patient's knee angle (KA) in this frame and the required 180 degrees. Error in a repetition e_{rep} is the average of e_{frame} among all the applied frames (i.e., frames of the applied sub-actions) in this repetition. The patient's overall error on this criterion is calculated as the mean and maximum of e_{rep} for all the repetitions. The mean and maximum error e_{mean} and e_{max} will be used as features of the task recommendation model, which will be discussed in Section III-D.

```
Algorithm 1: TPHAU algorithm
Input:    Two HMM models HMM-M and HMM-S,
          patient's displacement sequence $D = \{d_1, d_2, \ldots, d_T\}$
Output:   Segmentation of sub-actions
  Normalize $D$ into [0, 1] and calculate velocity $V$
  Apply Viterbi algorithm on $O = \{D; V\}$ using HMM-M,
  get the hidden state sequence $Q$
  for each consecutive $S_1$ sequence in $Q$
      Starting point of a repetition = midpoint of the
      consecutive $S_1$ sequence
  end for
  Detect extra repetitions using (4) and merge each of them
  into its previous or next repetition (whichever is closer)
  for each remaining repetition
      Normalize the displacement sequence of this repetition
      $D_{rep}$ into [0, 1], calculate $V_{rep}$
      Apply Viterbi algorithm on $O_{rep} = \{D_{rep}; V_{rep}\}$ using
      HMM-S, get the hidden state sequence $Q_{rep}$
      Segment sub-actions in this repetition based on $Q_{rep}$
  end for
```

Fig. 7. Pseudo-code of the proposed TPHAU algorithm.

In additional to the quantitative errors, qualitative assessments (i.e., the patient's performance is either satisfactory or non-satisfactory on a criterion) are also crucial in providing feedback for the patient. If the patient's performance on a criterion is non-satisfactory, guidance will be rendered on the cloud and sent to the user's device to instruct him/her to improve the performance. Therefore, we build an SVM-based classification model [25]. For each training sample, the mean and maximum errors on a criterion are used as the input feature. The label y of the sample is given by the PT based on the patient's performance during the data collection process, with $y = 1$ representing positive (i.e., the performance is satisfactory on this criterion) and $y = 0$ representing negative (i.e., the performance is non-satisfactory on this criterion). A linear binary SVM classifier is trained from the training data to find out the optimal decision boundary between the positive and negative samples. Since the criteria are independent of each other, a unique classification model is trained for each criterion of each task.

D. Machine Learning-Based Task Recommendation

In this section, we propose a task recommendation model to emulate the PT's decisions in updating the training tasks (i.e., the difficulty level for each task). Section III-D-1 introduces the input and output of the model. Section III-D-2discusses the imbalanced data problem and existing methods. In Section III-D-3, we propose a novel hybrid over-sampling approach to address the imbalanced data problem.

1) Task Recommendation Framework

To enable automated task update recommendation, we propose a random forest-based classification model to emulate the PT's decision in updating the difficulty level of each task based on the patient's performance. Random forest (RF) is an ensemble learning method for classification, regression, and other problems [26]. Output of the proposed model is the PT's decision in updating the difficulty level, which are quantified into three categories: ***Progress*** (i.e., from level k to k+1), ***Repeat*** (i.e., repeat the current level k), and ***Regress*** (i.e., from level k to k-1). Note that a patient cannot progress any more when the current level is 4, but the PT may still assign *progress* if his/her performance is excellent in order to help the model learn the difference between ordinary and excellent performance. For the current level 1, the difference between *repeat* and *regress* are also clarified although outcomes for both situations are level 1. Inputs/Features of the model include the patient's maximum and mean error on each criterion (discussed in Section III-C). Besides, some subjective factors (e.g., pain, age/sex, etc.) may also affect the PT's decision on task recommendation. For example, the PT may recommend *Repeat* to a patient with knee pain, even if the patient performs well on the current level. Table IV shows all the features used in the task recommendation model.

TABLE IV. FEATURES OF THE RF CLASSIFIER.

Type	**Feature**	**Value**
Continuous	Maximum/mean error on a criterion	Criterion-specific
	Age	56 ~ 89
Nominal	Sex	M/F
	Current difficulty level	1/2/3/4
	Knee pain	Y/N
	Back/hip pain	

2) Imbalanced Data Problem and Existing Methods

For the patient data that we have collected in the clinic, each sample (i.e., a patient performing a task once) belongs to one of the three categories (*Regress*, *Repeat*, *Progress*) based on the PT's recommendation on the task update. Table V shows the distribution of collected samples in the three classes for the three training tasks. We can see that the collected data are imbalanced for the three categories. The PT is conservative in regressing the patient, thus the percentage of samples in class *Regress* is very low (under 15%). As for *Repeat* and *Progress*, fewer patients (about only 20%) can progress to the next level for FL/BL than SQ. It may be because FL and BL are more challenging than SQ as they involve dynamic weight shift from on foot to the other, which is particularly difficult for patients with PD.

Because of the imbalanced data problem, the RF classifier may be biased towards the majority class (e.g., class *Repeat* for FL) to achieve high overall accuracy. For example, a classifier applied on a training dataset with 95% positive samples and 5% negative samples can achieve high overall accuracy of 95% by using the simple strategy of always predicting positive. However, the cost of misclassifying a minority sample as a majority sample can sometimes be much higher than the cost of the reverse error. For example, predicting a patient who should *Regress* to the lower level (due to severe pain or errors) as *Repeat* and *Progress* may cause injury to the patient. Therefore, we should focus on the accuracy of each individual class instead of the overall accuracy. Next, we describe techniques that have been developed to address the imbalanced data problem in other applications, point out issues in utilizing these techniques, and subsequently propose a new technique for our application. Results of using our proposed technique in comparison with the existing methods will be provided in Section IV-D.

TABLE V. SAMPLE DISTRIBUTION FOR SQUAT (SQ), FORWARD LUNGE (FL), BACKWARD LUNGE (BL).

Task	Class (PT recommendation)		
	Regress	*Repeat*	*Progress*
SQ	13.5%	42.5%	44.0%
FL	11.8%	67.8%	20.4%
BL	12.6%	63.2%	24.2%

Majority under-sampling [32]. It reduces the number of majority samples by selecting part of the majority samples. Because of the limited number of collected training samples in our task recommendation system, it may have negative effects on the accuracy.

Minority over-sampling with replacement [33]. It increases the number of minority samples by creating minority duplicates. However, Ling et al. [27] propose that it may cause over-fitting problem as it makes the decision region for the minority class more specific.

Decision threshold adjustment [31]. For a normal RF classifier, the probabilities of all the classes are calculated and the one with the highest probability is selected as final classification result. Provost et al. [31] propose to tune the decision boundary to be biased towards the minority class, which is equivalent to assigning larger weight on the probability of the minority class. For the PT task recommendation problem, we can assign weights to the predicted probabilities as $\{w_{reg}\cdot P(\mathit{Regress}), w_{rep}\cdot P(\mathit{Repeat}), w_{prog}\cdot P(\mathit{Progress})\}$ (w_{reg} may be greater than w_{rep} and w_{prog}) and then select the class with highest probability. However, Chawla et al. [28] has shown that simply changing the decision threshold cannot always guarantee better results.

Synthetic minority over-sampling [28]. The minority class is over-sampled by taking each minority sample and introducing synthetic samples between the sample and its nearest neighbors. The distance *dist* between two samples *A* and *B* is calculated as

$$dist = sqrt[\sum_{i=1}^{M}(A_{f_i} - B_{f_i})^2 + \sum_{j=M+1}^{M+N}\delta(A_{f_j}, B_{f_j})\cdot Med^2] \quad (5)$$

$$\delta(A_{f_j}, B_{f_j}) = \begin{cases} 0, & if\ A_{f_j} = B_{f_j} \\ 1, & otherwise \end{cases}, \quad (6)$$

where $\{f_1, \ldots, f_M\}$ are continuous features, $\{f_{M+1}, \ldots, f_{M+N}\}$ are nominal features, and *Med* is the median of standard deviations of all continuous features for the minority class. For continuous features, the Euclidean distance is included in *dist*. For nominal features, *Med* is included in *dist* if *A* and *B* have different values on this feature. For each minority sample, *k* nearest neighbors are found and *p* neighbors among them ($p \leq k$) are randomly selected, depending on the over-sampling rate $p\cdot100\%$. A synthetic sample is generated between the minority sample and each of the selected *p* neighbors. If *p* is not an integer, use *ceil*(*p*) first and randomly select a percentage of $p/ceil(p)\cdot100\%$ from all the synthetic samples. For continuous features f_c, linear interpolation is used to generate the new sample *C* as

$$C_{f_c} = A_{f_c} + m\cdot(B_{f_c} - A_{f_c}) \quad (7)$$

where *m* is a random number between 0 and 1. For nominal features, the value occurring in the majority of the *k* nearest neighbors is assigned to *C*. However, applying the traditional over-sampling method in our dataset does not give satisfying results. We will explain the problems and discuss the solutions in the next section.

3) *Proposed Hybrid Over-Sampling Approach*

Based on the traditional synthetic minority over-sampling method [28], we propose a novel hybrid over-sampling approach. In this section, we will first introduce a pre-processing step (feature standardization), and then introduce the problems of applying the traditional over-sampling method [28] in our dataset and discuss our proposed solutions.

a) *Feature standardization for continuous features*

The continuous features we use in the task recommendation model (see Table IV) use different units of measurement and differ greatly in value range. For example, the value range of the patient's age is 56 ~ 89 while the patient's error on the criterion "normalized length of step" has the value range of 0 ~ 0.3. Therefore, features with greater values will dominate in the distance calculation in (5) and features with smaller values may be ignored. To solve this problem, we propose a feature standardization step to preprocess the continuous features: all continuous features are normalized to zero-mean and unit variance before the distance calculation.

b) *Hybrid interpolation for error features*

There are some problems with the traditional linear interpolation approach (7) when generating synthetic samples. We will first use the error features (i.e., patient's error on each criterion) as an example to illustrate the problem and propose our solutions, then generalize the solutions to the other features. Table VI shows a simple example of the error features on two criteria C_1 and C_2. The PT recommends *Regress* for sample *A* and *B* for different reasons: *A*'s performance on both criteria are non-satisfactory (Non-Sat) and *B*'s error on C_2 is too large (50°). By using linear interpolation (with the random number $m = 0.5$ in (7)), the synthetic sample *C* has error = 10° on C_1 (which may be Sat) and error = 30° on C_2 (Non-Sat). However, the PT may use complicated strategies in making recommendations instead of simply counting the number of Sat criteria. For example, if a sample has one Sat and one Non-Sat for the two criteria, the PT may recommend *Regress* only if the error of Non-Sat is too large (e.g., 50° on C_2 for sample *B*). Therefore, the PT may not recommend *Regress* for sample *C* since its error on C_2 is not so important. To create a correct *Regress* sample, we first propose a biased interpolation method based on the following fact: a *Regress* sample will still be in class *Regress* if any/all of its error features get larger in value. For the example in Table VI, a synthetic sample *D* that uses larger value of *A* and *B* on each error feature must also be a *Regress* sample.

TABLE VI. DIFFERENT INTERPOLATION METHODS WHEN GENERATING SYNTHETIC SAMPLES

Sample	Error on C_1	Error on C_2	Recommendation
A	20° (Non-Sat)	10° (Non-Sat)	*Regress*
B	0° (Sat)	50° (Non-Sat)	*Regress*
C (linear)	10° (Sat)	30° (Non-Sat)	*Regress*?
D (biased)	20° (Non-Sat)	50° (Non-Sat)	*Regress*
E (hybrid)	20° (Non-Sat, biased)	30° (Non-Sat, linear)	*Regress*

However, using biased interpolation on all the error features may cause the synthetic samples to be too far away from the original minority samples and the decision boundary to be not optimal for the original minority samples. Fig. 8 shows an example of a majority class and a minority class. When the synthetic samples are far away from the original samples (see (a)), the decision boundary causes a high error rate on the original minority samples. To achieve the optimal over-sampling results, the synthetic samples should be among the original minority samples (as shown in (b)).

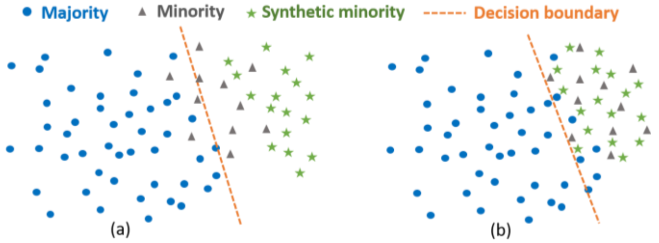

Fig. 8. Minority over-sampling. (a) Synthetic samples are far away from original minority samples. (b) Synthetic samples are among original minority samples.

Therefore, we propose a hybrid interpolation approach to create synthetic over-sampling samples. When generating a synthetic *Regress* sample *E* from original *Regress* samples *A* and *B*, the SVM classifier (discussed in Section III-C) is applied on *A* and *B* for each error feature. If both *A* and *B* are classified as Sat (or both are Non-Sat) on a criterion, linear interpolation in (7) is used to create the value of the synthetic sample. Otherwise, biased interpolation will be used (i.e., use the larger error value of *A* and *B*). The last row in Table VI illustrates this approach.

c) *Hybrid interpolation: generalization to the other features*

To generalize the proposed hybrid interpolation approach to the other features, we define features with Clear Effects on Performance (CEoP) as those features which can cause patient's better/worse performance, e.g., knee pain will cause worse performance. However, age and sex have no clear/direct effects on the performance. Based on the definition, we propose different interpolation methods for different features as follows. **(i) Continuous features w/ CEoP**: including patient's error on each criterion. Hybrid interpolation described in the previous section is used. **(ii) Continuous features w/o CEoP**: age. Since it has no clear effects on the performance, the proposed biased and hybrid interpolation approach cannot be used. Thus, we use linear interpolation on it. **(iii) Nominal features w/ CEoP**: including knee pain and back/hip pain. As linear interpolation (which is part of the proposed hybrid approach) cannot be used on nominal features, we use biased interpolation: if the two *Regress* samples differ in value (one is Y and the other is N), use Y for the synthetic *Regress* sample. **(iv) Nominal features w/o CEoP**: including sex and current difficulty level. Both biased and linear interpolation cannot be used on it. Hence, we use the value occurring in the majority of the *k* nearest neighbors for the synthetic *Regress* sample. The pseudo-code for the proposed hybrid interpolation approach is shown in Fig. 9.

```
Algorithm 2: Hybrid over-sampling (for Regress samples)
Input:   Regress samples A and B
Output:  Synthetic Regress sample C
for each feature f of C
    if f is continuous feature w/ CEoP
        Apply the SVM-based error identification model
          on $A_f$ and $B_f$, get the prediction results $p_A$ and $p_B$
        if $p_A$ equals $p_B$
            $C_f = A_f + m \cdot (B_f - A_f)$
        else
            $C_f = \max(A_f, B_f)$
        end if
    else if f is a continuous feature w/o CEoP
        $C_f = A_f + m \cdot (B_f - A_f)$
    else if f is a nominal feature w/ CEoP
        if $A_f$ equals $B_f$
            $C_f = A_f$
        else
            $C_f = Y$
        end if
    else
        $C_f$ uses the value occurring in the majority of the
          k nearest neighbors of A on feature f
    end if
end for
```

Fig. 9. Pseudo-code of the hybrid over-sampling approach.

For class *Progress*, the same hybrid interpolation approach can be used for synthetic over-sampling except that a smaller error value and pain = N will be used in biased interpolation. For class *Repeat*, biased interpolation cannot be used since it is an intermediate class. From Table V, we can see that class *Repeat* is not a minority class for all the three tasks discussed in this paper, thus over-sampling is not needed for it.

IV. Results

In this Section, we will first introduce the data collection process, and then present the results of the proposed patient action understanding, error identification, and task recommendation models. We also analyze and report the runtime efficiency of the proposed algorithms in the Appendix.

A. *Experimental Setup and Data Collection*

This research was approved by the Institutional Review Board at UC San Diego (protocol #181413X). 35 patients with PD (age 56 ~ 89, 22 males, 13 females, Hoehn & Yahr stage 1 ~ 4) recruited from the Neurological Rehabilitation Clinic, UC San Diego Health, participated in the data collection. All subjects signed the informed consent form. Each patient participated in the data collection for multiple times. Patient's motion data were recorded by a Microsoft Kinect v2 sensor. The corresponding PT assessments (i.e., whether the patient's performance was satisfactory or not on each criterion) and recommendations (i.e., *regress*, *repeat* or *progress*) were also recorded. For each task, the motion of one patient in one session constitutes a data sample. Each patient participated in the data collection for 2 ~ 4 times. Note that sometimes some patients were not able to perform some tasks (e.g., BL was too difficult for some patients), thus the number of collected samples for each task were different. We collected 96 samples for SQ, 93 samples for FL, and 87 samples for BL in total. Typically, patient's movements on a task includes 4 repetitions, with about 10 seconds on each repetition. The Kinect sensor captures the (x, y, z) coordinates of 25 joints per frame. With frame rate of 30 frames/second, that amounts to about 90,000 data points for each task performed by a patient in one session.

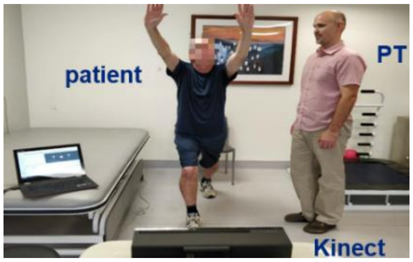

Fig. 10. Data collection in PT clinic.

B. *Patient Action Understanding Results*

To validate the proposed TPHAU algorithm, we conduct experiments using the stratified 10-fold cross validation on SQ, FL, BL separately, with 90% of the samples for each task used for training and 10% for test. The comparison between the one-phase Viterbi algorithm [23] and the proposed TPHAU algorithm is shown in Table VII. For repetition detection, the percentage of correct, wrong, missing repetitions, and the number of extra repetitions (discussed in Section III-B) are calculated. We can see that the proposed TPHAU algorithm enhances the accuracy of repetition detection significantly, with more correct repetitions and much less extra repetitions, especially for BL. For sub-action segmentation, we evaluate the

TABLE VII. REPETITION DETECTION AND SUB-ACTION SEGMENTATION RESULTS FOR SQUAT (SQ), FORWARD LUNGE (FL), BACKWARD LUNGE (BL)

Method	**Task**	**Repetition detection**				**Sub-action segmentation: Sensitivity**			**Sub-action segmentation: Specificity**		
		Correct	**Wrong**	**Missing**	**No. of extra repetitions**	S_2	S_3	S_4	S_2	S_3	S_4
One-phase Viterbi [23]	SQ	90.6%	9.4%	0%	5	91.1%	78.3%	92.8%	93.6%	97.9%	94.2%
	FL	97.9%	2.1%	0%	14	92.8%	86.2%	92.1%	96.2%	98.0%	94.1%
	BL	96.3%	3.7%	0%	56	92.9%	81.0%	87.6%	92.1%	97.5%	96.7%
TPHAU (proposed)	SQ	97.1%	2.9%	0%	0	89.5%	94.4%	90.8%	97.3%	98.1%	98.2%
	FL	97.9%	2.1%	0%	2	93.5%	96.4%	92.5%	97.9%	98.2%	97.2%
	BL	99.4%	0.6%	0%	6	92.0%	96.9%	88.4%	98.0%	97.5%	98.8%

TABLE VIII. ACCURACY OF ERROR IDENTIFICATION MODELS FOR SQUAT (SQ), FORWARD LUNGE (FL), BACKWARD LUNGE (BL).

Criterion for SQ	**Accuracy**	**Criterion for FL**	**Accuracy**	**Criterion for BL**	**Accuracy**
Sit hips back towards a chair	92.5%	Keep the back knee straight	86.3%	Keep the back knee straight	93.5%
		Keep the posture tall	93.2%	Keep the trunk parallel with the back leg	90.2%
Lean forward	89.1%	Length of step	93.8%	Length of step	94.2%
		Keep the front shank vertical	91.1%	Keep the front shank vertical	88.7%

TABLE IX. ACCURACY AND THE FALSE POSITIVE RATE (FPR) OF THE TASK RECOMMENDATION MODELS USING DIFFERENT METHODS FOR THE IMBALANCED DATA PROBLEM, FOR SQUAT (SQ), FORWARD LUNGE (FL), BACKWARD LUNGE (BL).

Method	**SQ**				**FL**				**BL**			
	Accuracy			**FPR**	**Accuracy**			**FPR**	**Accuracy**			**FPR**
	Regress	***Repeat***	***Progress***	***Repeat***	***Regress***	***Repeat***	***Progress***	***Repeat***	***Regress***	***Repeat***	***Progress***	***Repeat***
Original	69.2%	91.7%	95.7%	2.8%	54.5%	88.1%	76.5%	3.2%	63.6%	92.7%	85.0%	3.6%
Under-Samp [32]	84.6%	75.0%	87.2%	11.1%	81.8%	81.0%	73.7%	9.5%	81.8%	67.2%	85.0%	7.3%
Over-Repl [33]	76.9%	88.9%	95.7%	11.1%	63.6%	85.5%	75.0%	9.7%	63.6%	92.7%	85.0%	3.6%
Over-Synth [28]	69.2%	88.3%	95.7%	11.1%	72.7%	83.9%	85.0%	8.1%	72.7%	89.1%	90.0%	3.6%
Thold-Adj [31]	92.3%	86.1%	91.5%	5.6%	81.8%	68.3%	84.2%	7.9%	81.8%	72.7%	90.0%	7.3%
Proposed	92.3%	88.9%	95.7%	2.8%	81.8%	85.7%	84.2%	3.2%	81.8%	90.9%	90.0%	5.4%
No Subjective factors	76.9%	86.1%	95.7%	11.1%	72.7%	85.7%	84.2%	6.3%	27.3%	76.4%	90.0%	7.3%

accuracy of each sub-action S_2/S_3/S_4 separately. (S_1 is not evaluated since it is not important for the patient's performance.) For sub-action S_2/S_3/S_4, the sensitivity and specificity are shown in Table VII. We can see that the proposed TPHAU algorithm improves both sensitivity and specificity for the three tasks. Especially for sensitivity, TPHAU enhances the sensitivity for S_3 significantly (e.g., from 78.3% to 94.4% for SQ). For S_2 and S_4, the average sensitivity of TPHAU is sometimes slightly lower than the one-phase Viterbi. For example, the sensitivity of S_2 in SQ is 89.5% using TPHAU, which is slightly lower than the sensitivity of 91.1% achieved by the one-phase Viterbi method. However, the small difference may be due to the PT's subjective bias when manually segmenting the states. Therefore, we can conclude that the overall accuracy of the proposed TPHAU algorithm outperforms the one-phase Viterbi method.

C. Patient Error Identification Results

To validate the SVM-based patient error identification method, we use the same training/test set as Section IV-B. A linear SVM classifier is trained for each criterion. The accuracy of each criterion is calculated as the ratio of the correctly classified samples to the total number of test samples. Table VIII shows the results for the three tasks. For most criteria, the accuracy is above 90%. For two criteria "Lean forward" in SQ and "Keep the front shank vertical" in BL, the accuracy is close to 90%. For only one criterion "Keep the back knee straight" in FL, the accuracy is 86.3%. Hence, it is reasonable to conclude that the SVM-based model can provide accurate error identification.

D. Task Recommendation Results

To validate the proposed task recommendation approach, we build three RF-based task recommendation models for SQ, FL, BL separately. To solve the imbalanced data problem, we apply the techniques introduced in Section III-D-2: majority under-sampling (***Under-Samp***) [32], minority over-sampling with replacement (***Over-Repl***) [33], traditional synthetic minority over-sampling using linear interpolation (***Over-Synth***) [28], decision threshold adjustment (***Thold-Adj***) [31], and the proposed hybrid synthetic over-sampling approach (***Proposed***). For under-sampling, the majority classes *Repeat* and *Progress* are under-sampled to a similar size of the minority class *Regress*. For over-sampling, class *Regress* is over-sampled to a similar size of class *Repeat*. Since class *Progress* also has less samples than class *Repeat* for FL/BL, we apply slight over-sampling on class *Progress*. (Between *Progress* and *Repeat*, a slight bias towards *Repeat* is preferred as the cost of misclassifying a *Progress* sample as *Repeat* is just delaying the patient's progress while the reverse error may cause health risks. Therefore, we apply slight instead of ordinary over-sampling on class *Progress*. Slight over-sampling means smaller over-sampling rate and fewer synthetic samples are created compared with ordinary over-sampling.) The accuracy of each class is calculated. Besides, the accuracy of class *Repeat* is affected by two types of errors: A) misclassifying *Repeat* as *Regress* (which may delay patient's progress), and B) misclassifying *Repeat* as *Progress* (which may cause risks). We consider the type B error more harmful, thus we also calculate the type B error as the False Positive Rate (FPR) of class *Repeat*.

The original results (without using any method to solve the imbalanced data problem) and results by using these techniques are shown in Table IX. For the original imbalanced dataset, the majority class (i.e., *Repeat* and *Progress* for SQ, *Repeat* for FL and BL) achieves high accuracy (around 90%) while the accuracy of the minority class *Regress* is much lower (below 70%). Among all the methods, ***Over-Repl*** and ***Over-Synth*** are not able to improve the accuracy of class *Regress* significantly. The two methods ***Under-Samp*** and ***Thold-Adj*** increase the accuracy of class *Regress*, however with the cost of high FPR of class *Repeat* (e.g., FPR of *Repeat* is 9.5% using ***Under-Samp*** and 7.9% using ***Thold-Adj*** for FL). Overall, our proposed hybrid synthetic over-sampling approach outperforms the other methods in increasing the accuracy of the minority class while maintaining high accuracy and low FPR on the majority class.

To show the importance of including the subjective factors (discussed in Section III-D-1) in the PT's recommendation, we conduct experiments by removing all the subjective factors from the features and applying the proposed hybrid over-sampling approach. Results are shown in the last row of Table IX. We can see that accuracy drops significantly, especially for class *Regress*. It is reasonable since some of the subjective factors (e.g., knee pain) indicate patient's poor health condition, which may be the primary reason of PT's decision to regress the patient.

V. Discussion

In this paper, we propose a virtual PT system to enable on-demand remote training for patients with PD. Patient's movements can be understood by the proposed TPHAU algorithm and errors are identified by SVM-based models. To enable automated task recommendation, a machine learning-based model is developed and trained from real patient data, which can emulate the human PT's recommendations. Experiments on patient data show that the proposed methods can accurately understand the patient's actions, identify errors, and provide task recommendation like a real PT. The proposed virtual PT system has the potential of enabling on-demand virtual care and significantly reducing cost for both the patients and care providers.

In the future, we plan to incorporate other kinds of sensors, like pressure sensors and epidermal sensors, in the training systems. Furthermore, the proposed virtual PT system can be generalized to other diseases (e.g., stroke) by designing the disease-specific training tasks and criteria. However, more issues need to be discussed. For example, the detection accuracy of Kinect may degrade when tracking more complicated movements or patients with walkers and wheelchairs. Besides, there might be more advanced progress manners (e.g., progress from level 2 to level 4) for some training tasks. All these issues will be considered and explored in our future work.

Acknowledgment

This material is based upon work supported by the National Science Foundation under grant No. IIS-1522125.

References

[1] Parkinson's disease statistics by Parkinson's Foundation. [Online]. Available: http://parkinson.org/Understanding-Parkinsons/Causes-and-Statistics/Statistics

[2] L. A. King, et al., "Effects of Group, Individual, and Home Exercise in Persons With Parkinson Disease: A Randomized Clinical Trial," *Journal of neurologic physical therapy: JNPT* 39.4 (2015): 204-212.

[3] M. D. Hssayeni, J. L. Adams, and B. Ghoraani, "Deep Learning for Medication Assessment of Individuals with Parkinson's Disease Using Wearable Sensors," In Proc. of *40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (EMBC'18), Honolulu, HI, USA, July 2018.

[4] E. Stack, et al. "Identifying balance impairments in people with Parkinson's disease using video and wearable sensors," Gait & posture 62 (2018): 321-326.

[5] B. Galna, et al., "Retraining function in people with Parkinson's disease using the Microsoft kinect: game design and pilot testing," *Journal of neuroengineering and rehabilitation* 11.1 (2014): 60.

[6] J. E. Pompeu, et al., "Feasibility, safety and outcomes of playing Kinect Adventures!™ for people with Parkinson's disease: a pilot study," *Physiotherapy* 100.2 (2014): 162-168.

[7] W. Wei, Y. Lu, E. Rhoden, and S. Dey, "User performance evaluation and real-time guidance in cloud-based physical therapy monitoring and guidance system," *Multimedia Tools and Applications* (2017): 1-31.

[8] Kinect. [Online]. Available: www.xbox.com/en-US/kinect

[9] Z. Wang, J. Liao, Q. Cao, H. Qi, and Z. Wang, "Friendbook: a semantic-based friend recommendation system for social networks," *IEEE transactions on mobile computing* 14.3 (2015): 538-551.

[10] M. Yan, J. Sang, and C. Xu, "Unified youtube video recommendation via cross-network collaboration," in Proc. of *the 5th ACM on International Conference on Multimedia Retrieval*(ICMR'15), Shanghai, China, Jun. 2015.

[11] W. Wei, C. McElroy, and S. Dey, "Human Action Understanding and Movement Error Identification for the Treatment of Patients with Parkinson's Disease," in Proc. of *IEEE International Conference on Healthcare Informatics* (ICHI'18), New York City, USA, Jun. 2018.

[12] B. Galna, et al., "Accuracy of the Microsoft Kinect sensor for measuring movement in people with Parkinson's disease," *Gait & posture* 39.4 (2014): 1062-1068.

[13] T. Y. Lin, C. H. Hsieh, and J. D. Lee, "A kinect-based system for physical rehabilitation: Utilizing tai chi exercises to improve movement disorders in patients with balance ability," in Proc. of *the 2013 7th Asia Modelling Symposum* (AMS'13), Hong Kong, China, Jul. 2013.

[14] L. Xia, C. C. Chen, and J. K. Aggarwal, "View invariant human action recognition using histograms of 3d joints," in Proc. of *the 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops* (CVPRW'12), Providence, RI, USA, Jun. 2012.

[15] J. Sung, C. Ponce, B. Selman, and A. Saxena, "Unstructured human activity detection from rgbd images," in Proc. of *the 2012 IEEE International Conference on Robotics and Automation* (ICRA'12), Saint Paul, MN, USA, Jun. 2012.

[16] H. Pirsiavash, and D. Ramanan. "Parsing videos of actions with segmental grammars," in Proc. of *the 2014 IEEE Conference on Computer Vision and Pattern Recognition* (CVPR'14), Columbus, OH, USA, Jun. 2014.

[17] C. Wu, J. Zhang, S. Savarese, and A. Saxena, "Watch-n-patch: Unsupervised understanding of actions and relations," in Proc. of *the 2015 IEEE Conference on Computer Vision and Pattern Recognition* (CVPR'15), Boston, MA, USA, Jun. 2015.

[18] L. A. King, F. B. Horak, "Delaying mobility disability in people with Parkinson disease using a sensorimotor agility exercise program," *Physical Therapy* 89.4 (2009): 384-393.

[19] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE* 77.2 (1989): 257-286.

[20] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, “A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains,” *The annals of mathematical statistics* 41.1 (1970): 164-171.

[21] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi, “An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition,” *The Bell System Technical Journal* 62.4 (1983): 1035-1074.

[22] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of the royal statistical society. Series B (methodological)* (1977): 1-38.

[23] G. D. Forney, “The viterbi algorithm,” *Proceedings of the IEEE* 61.3 (1973): 268-278.

[24] F. Pukelsheim, “The three sigma rule,” *The American Statistician* 48.2 (1994): 88-91.

[25] C. J. Burges, “A tutorial on support vector machines for pattern recognition,” *Data mining and knowledge discovery* 2.2 (1998): 121-167.

[26] L. Breiman, “Random forests,” *Machine learning* 45.1 (2001): 5-32.

[27] C. X. Ling, and C. Li, “Data mining for direct marketing: Problems and solutions,” *KDD*. Vol. 98. 1998.

[28] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: synthetic minority over-sampling technique,” *Journal of artificial intelligence research* 16 (2002): 321-357.

[29] Z. Yu, C. Wang, J. Bu, X. Wang, Y. Wu, and C. Chen, “Friend recommendation with content spread enhancement in social networks,” *Information Sciences* 309 (2015): 102-118.

[30] Y. Deldjoo, M. Elahi, P. Cremonesi, F. Garzotto, P. Piazzolla, and M. Quadrana, “Content-based video recommendation system based on stylistic visual features,” *Journal on Data Semantics* 5.2 (2016): 99-113.

[31] F. Provost, and T. Fawcett, “Robust classification for imprecise environments,” *Machine learning* 42.3 (2001): 203-2.

[32] M. Kubat, and S. Matwin, ”Addressing the curse of imbalanced training sets: one-sided selection,” *Icml*. Vol. 97. 1997.

[33] C. X. Ling, and C. Li, “Data mining for direct marketing: Problems and solutions,” *Kdd*. Vol. 98. 1998.

Appendix: Runtime Efficiency of the Proposed Algorithms

For the three algorithms proposed in this paper, we have conducted comprehensive experiments to test their runtime efficiency. The running time of each algorithm is tested on an Intel Xeon E5-1650 CPU. For each algorithm, there is an offline training stage (in which the model is trained on training samples) and a test/inference stage (in which the trained model is applied on new/test samples). Since the proposed virtual PT system is cloud-based, high efficiency is needed in the inference stage to provide action understanding, error identification, and task recommendation in a timely manner. Therefore, the runtime efficiency in the inference stage is of greater importance. In this appendix, we will present the running time of each algorithm in both training and inference stages.

A. *The TPHAU algorithm*

It is compared with the one-phase Viterbi algorithm. The running time of the training and inference stage is shown in Table X. The total training time is the total time taken to train the model on all the training samples. The average inference time is the average running time of applying the model on a new/test sample. Since the two algorithms differ only in the reference stage, their training time is the same (about 20 *s* for each task). In the inference stage, the proposed TPHAU algorithm requires more running time due to the use of the second phase to improve the detection accuracy (discussed in Section III-B). From Table X we can see that it takes less than 150 *ms* to apply the proposed TPHAU algorithm on a new/test sample in the inference stage, which means that action understanding can be performed in real time.

TABLE X. Running time of the One-Phase Viterbi Algorithm and the Proposed TPHAU Algorithm, for Squat (SQ), Forward Lunge (FL), and Backward Lunge (BL).

Method	**Total Training Time (*s*)**			**Average Inference Time (*ms*)**		
	SQ	***FL***	***BL***	***SQ***	***FL***	***BL***
One-phase Viterbi [23]	17.3	21.3	20.2	65.4	102.2	111.8
Proposed TPHAU				81.9	127.5	139.5

B. *The error identification model*

For error identification, a SVM classifier is used to identify whether the patient's performance is satisfactory or not on a PT-defined criterion. Since multiple criteria have been defined for each task (discussed in Section III-A), the running time of each criterion is summed up as the total running time needed to evaluate the patient's performance on all PT-defined criteria for this task. We summarize the running time in both training and inference stage in Table XI. We can see that the training stage requires less than 30 *ms* for each task. The inference stage is very fast, requiring less than 0.1 *ms* for each task.

TABLE XI. Running time of the Proposed Error Identification Model, for Squat (SQ), Forward Lunge (FL), and Backward Lunge (BL).

Method	**Total Training Time (*ms*)**			**Average Inference Time (*ms*)**		
	SQ	***FL***	***BL***	***SQ***	***FL***	***BL***
Proposed SVM model	14.9	27.4	27.1	0.02	0.04	0.05

C. *The task recommendation model*

The proposed task recommendation model is based on the random forest classifier. Because of the imbalanced data problem (discussed in Section III-D-2), we have proposed the hybrid synthetic over-sampling approach to generate synthetic samples for the minority class in the training stage and have shown its results compared with other methods (discussed in Section III-D-2 and Section IV-D). Table XII shows the total training time required by each method for the imbalanced data problem. We can see that the training time of the traditional synthetic over-sampling approach (Over-Synth) and the proposed hybrid synthetic over-sampling approach (Proposed) is higher than the other techniques because these two methods requires extra steps to generate the new synthetic samples. For the inference stage, the average inference time is the same for all the methods since these methods are applied only in the training stage to address the imbalanced data problem. We can see that the inference stage of the task recommendation model requires only 4 *ms* for each task.

TABLE XII. Running time of the Task Recommendation Models Using Different Methods for the Imbalanced Data Problem, for Squat (SQ), Forward Lunge (FL), and Backward Lunge (BL).

Method	**Total Training Time (*s*)**			**Average Inference Time (*ms*)**		
	SQ	***FL***	***BL***	***SQ***	***FL***	***BL***
Original	2.9	3.2	3.1	3.9	4.0	4.1
Under-Samp [32]	2.8	3.0	2.8			
Over-Repl [33]	2.9	3.2	3.0			
Over-Synth [28]	6.1	13.5	11.2			
Thold-Adj [31]	2.9	3.1	3.1			
Proposed	6.6	15.9	14.0			
No Subjective factors	5.1	10.0	9.0			

From the results presented above, we can see that the running time of the three proposed models (i.e., the TPHAU algorithm for patient action understanding, the SVM-based error identification model, and the task recommendation model) in the inference stage is about 150 *ms* in total. It means that the virtual PT system can evaluate the patient's performance and provide task recommendation in about 150 *ms* after the patient completes a training task, which enables efficient and real-time remote care.