Interpretable Models for Fast Activity Recognition and Anomaly Explanation During Collaborative Robotics Tasks

Bradley Hayes[1] and Julie A. Shah[1]

Abstract— **In this paper, we present Rapid Activity Prediction Through Object-oriented Regression (RAPTOR), a scalable method for performing rapid, real-time activity recognition and prediction that achieves state-of-the-art classification accuracy on both a generic human activity dataset and two domain-specific collaborative robotics manufacturing datasets. Our approach is designed to be human-interpretable: able to provide explanations for its reasoning such that non-experts can better understand and improve its activity models. We incorporate methods to increase RAPTOR's resilience against confusion due to temporal variations, as well as against learning false correlations between features. We report full and partial trajectory classification results across three datasets and conclude by demonstrating our model's ability to provide interpretable explanations of its reasoning using outlier detection techniques.**

I. Introduction

Activity recognition is a fundamental research challenge within the fields of robotics and computer vision, with potentially wide ranging impact on human-robot interaction, multi-agent systems, security surveillance, home automation, and manufacturing. In human-robot collaboration, the ability of a team member to quickly and reliably interpret teammates' actions and intentions is critical to achieving satisfactory robot performance and team fluency. This type of anticipatory information can improve task completion time, idle time, concurrent motion, and human-robot separation distance during human-robot collaboration [1], [2], [3], [4], [5], [6], [7]. Thus, rapid classification is a key attribute of an effective approach to activity recognition, as it allows for real-time situational awareness at the speed of the available sensing hardware. However, activity recognition presents substantial challenges that must be addressed before robust human-robot teaming can be realized in practice.

Although a variety of innovative machine learning approaches have been proposed to support activity recognition during human-robot collaboration [8], [9], [10], [11], the reasoning behind these classifiers' decisions is not necessarily interpretable to a human partner. With these methods a human may be able to inspect individual activity classification likelihoods and consider them relative to one another, but he or she would not be able to identify which specific aspects of a demonstration were responsible for the misclassification without substantial additional effort and a background in machine learning.

This poses challenges for effective communication of the robot's progress and intent, which can impede the development of shared mental models among team members [12], [13], [14]. With process efficiency and safety at stake during live task co-execution [15], [16], it is a priority that the mechanisms affecting the behavior and control of autonomous systems be transparent to human co-workers. To this end, we present a novel approach to activity recognition during collaborative robotics tasks, designed to provide both rapid online action classification and human-interpretable justifications for decisions and reasoning.

Our work introduces a scalable mechanism for rapid, online activity recognition that achieves state-of-the-art results for both general and domain-specific datasets, enabling a robot to quickly and accurately plan for and around the behaviors of its co-workers. The basis of our approach is the use of classifier ensembles for each activity, wherein we form and combine hypotheses from a collection of independent multivariate Gaussian Mixture Models that characterize the motion patterns of individual objects (e.g., joints) in a given scene. We construct these object-oriented motion models within a series of time windows over the provided training trajectories to capture local patterns in the data while minimizing false correlations between objects. We also introduce a modified form of max-pooling [17] to further increase resilience to temporal misalignment. Our method performs automated feature selection within each activity's classifier ensemble in order to identify the most relevant and informative object models for each time window.

The object-oriented nature of our approach facilitates anomaly detection and diagnosis, enabling each activity model to provide its own description of differences or similarities to queried activity demonstrations. To mitigate scaling issues for training and testing, our classifier architecture is highly parallel, allowing for expedited training times and rapid evaluation, even when considering between a large variety of activity classes.

Our proposed method demonstrates favorable generalization across variations within activities, exhibiting cutting-edge performance on cross-subject evaluations, where classifiers are tested on individuals they have not been trained on. Our approach exhibits resilience against misclassifications due to similarities across activities, placing greater weight on features that differentiate similar activities from one another. We maintain a robustness to time variance within activity demonstrations, enabling our classifier to overcome ambiguities arising from temporal inconsistencies such as delays or hurried demonstrations, while preserving intermediate pose-ordering information such as recognizing that a seated pose was held prior to a standing position. Operating solely on

[1]Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology, 32 Vassar St, Cambridge, MA 02139 {hayesbh, julie_a_shah}@csail.mit.edu

skeletal and object position data, our approach achieves high classification accuracies even in the absence of environment descriptors typically embedded within the RGB(+D) camera imagery that often accompanies such features [9], allowing us to minimize bandwidth requirements when performing activity recognition using distributed computation.

In this work, we evaluate our method in both live and offline settings across three activity datasets covering a wide range of behaviors. We conclude by showcasing the interpretable aspects of our classifier, demonstrating its ability to summarize anomalies within its input data in a human-interpretable manner.

II. Related Work

Activity recognition is commonly discussed in literature as a process of pose discovery – learning the most informative intermediate positions occupied during an activity – followed by a process of sequence classification over the recognized poses from this set. Defining activities explicitly as an evolution of spatial features (pose configurations) irrespective of relative timing introduces complexity with respect to handling intra-activity temporal disparities. To overcome this, state-of-the-art classifiers introduce novel methods of modeling the interplay between spatial and temporal aspects of activities. In this section, we review works that have incorporated skeletal and/or object position data – in particular, recent contributions that utilize both implicit and explicit approaches to the pose discovery problem.

A number of promising results have emerged through clustering-based pose discovery. Xia et al. [11] introduced the histograms of 3D joint locations (HOJ3D) method of activity recognition. In their approach, intermediate activity poses are found by re-projecting clustered joint locations using latent Dirichlet analysis, which are then modeled sequentially by activity-specific hidden Markov models (HMMs). Their work introduced and was evaluated against the UTKinect activity dataset, a standard activity recognition benchmark we use here as a point of comparison for our own results. Chrungoo et al. [18] also utilized histograms, grouping direction vectors from fixed temporal windows. Gaglio et al. [19] took a k-means clustering approach to intermediate pose discovery, sampling from subsets of training data to determine key pose configurations. In this work, the authors use a support vector machine (SVM) to find optimally separated poses in order to minimize inter-activity confusion.

Other authors have contributed high performing work that implicitly encodes key pose information. For example, Devanne et al. [20] modeled skeletal pose transitions as curves through a Riemannian manifold. In this work, activity trajectories are projected into a higher dimensional space, where an elastic metric used to compare various curve shapes is combined with a k-nearest neighbor classification strategy. A similar approach was pursued by Slama et al. [21], who classified activities with a linear SVM over bundles of vectors tangent to a given trajectory's position on a learned Grassmann manifold. In this approach, temporal modeling and classification are performed using a combination of time warping, a Fourier temporal pyramid trajectory representation, and a linear SVM classifier.

Presti et al. [22] used linear time invariant (LTI) systems to model subsets of activity demonstrations within fixed size sliding temporal windows, utilizing trained HMMs on the global space of LTIs for classification. A sliding window technique was also used by Gori et al. [9], who applied learned 1-D Gaussian filters through time over feature patches extracted from interactions between joint pairs, allowing for quick and robust identification of individual activities and multi-party interactions. Deep convolutional neural networks have also been applied within this domain, utilizing depth and motion information alongside feature transformations that convert the problem of activity recognition into one more closely resembling static image classification [23].

Pérez-D'Arpino et al. [24] also utilize a Gaussian method, implicitly modeling intermediate poses with multivariate Gaussian distributions over feature means at each time step. The authors used dynamic time warping (DTW) to achieve a canonical length for each activity, alleviating issues stemming from temporal deviations. They evaluated their classifier in a live human-robot collaborative setting, demonstrating rapid, accurate activity prediction during human reaching tasks using only small portions early in the segmented activity trajectory, enabling a robot to accurately anticipate human movement.

The lack of human-interpretable, online activity classifiers represents a key gap in the existing literature. Within the scope of human-robot collaboration, the ability of a human to gain insight into a robot collaborator's model of an activity can have important implications for worker safety and team fluency. When applied to active learning, interpretable models could allow for the presentation of more informative training examples, leading to improved task performance [25]. Unlike the ensembles of position-based models we present here, techniques that utilize dimensionality reduction or re-projection onto high dimensional geometric manifolds sacrifice the ability to directly summarize models into representations intuitive for humans. With respect to online classification, only Pérez D'Arpino et al. [24] and Slama et al. [21] address activity prediction, performing activity recognition using only a partial trajectory. In this work, we present best-in-class results in online classification with a human-interpretable approach that generalizes to more diverse activity classes than prior work while maintaining real-time performance on commodity computing hardware.

III. Method

Here, we introduce RAPTOR (Rapid Activity Prediction Through Object-oriented Regression), a highly parallel classifier for online activity recognition that achieves state-of-the-art results with both general and domain-specific datasets. Our approach allows for rapid training and testing, as each activity model and its constituent components can be trained and tested independently. Our proposed classifier

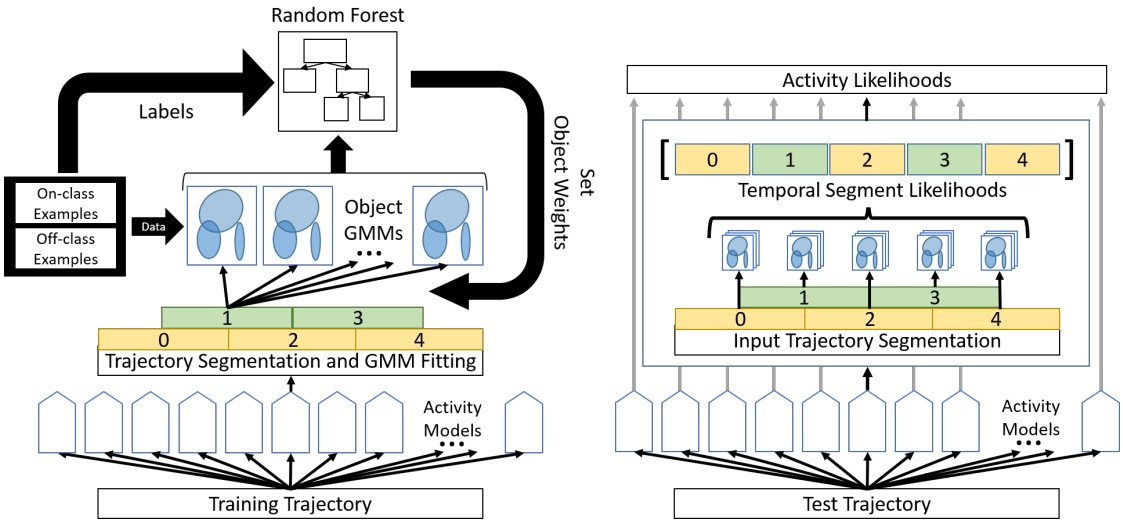

Fig. 1: (Left) A sketch of the training pipeline for RAPTOR classifiers. Trajectories are segmented into temporal windows within each activity model, where GMMs are fit for each object (feature subset). Object GMMs are weighted within each temporal segment through a Random Forest-driven feature selection process. (Right) A sketch of the RAPTOR classification pipeline. Input trajectories are passed to each activity model, where they are segmented according to the model's temporal window parameters, scored per segment by object GMMs, and combined into a single likelihood score per activity.

operates effectively on derived skeletal and position features (without the source image or point cloud), minimizing the bandwidth required for operation at scale over a large computing hardware network. Drawing inspiration from the success of Object Oriented Markov Decision Processes [26] in reinforcement learning, we factor our input feature space into 'objects' as a mechanism for improving classifier generalization and avoiding false correlations between potentially unrelated features within the training data.

Our algorithm performs activity classification on an input trajectory segment presented as an $N \times D$ input matrix with row vectors of D floating point values for N time steps. The object-oriented approach we have taken factors input matrices into feature collections per a mapping function specified at initialization of the form f : object $\rightarrow$ feature indices. For example, the UT-Kinect [11] dataset contains trajectories of 60 features per time step, belonging to 20 joints. On this dataset, our algorithm isolates sets of column vectors from the input matrices using masks corresponding to the position feature indices (X,Y,Z coordinates) of each joint, transforming the input trajectory from a single $N \times 60$ matrix to 20 non-overlapping $N \times 3$ matrices. This yields a collection of objects that independently represent each joint's motion over time. In a smart factory, these input features may resemble tracking markers from a motion capture system for each worker and parts bin located on the factory floor. This object segmentation affords our classifier benefits for both generalization and interpretation.

Given a motion trajectory, RAPTOR outputs a vector of classification likelihoods for each known activity class. Unlike the classification approaches referenced in Section II, our method also affords the capability to query each activity model for an interpretable explanation of anomalies in the input signal (e.g., "The beginning of the provided trajectory fits my model for 'install widget' exceptionally well, but for most of the end of the trajectory, the features associated with widget_position are very unfamilar."). In the remainder of this section, we present details about our classifier's architecture, training process, classification pipeline, and anomaly explanation capability.

A. *Classifier Architecture*

Each activity model within RAPTOR is trained independently as a collection of temporally segmented ensembles of Gaussian Mixture Models (GMMs), as shown in Figure 1. Each temporal segment contains its own ensemble of classifiers, locally modeling sequences of the object motions most relevant for distinguishing its target activity from other activities. A weighted combination of likelihood estimations from each classifier is computed at each time segment to generate a net likelihood measurement for the activity class. The maximum likelihood estimate over all activity models is then subsequently chosen as the predicted activity class (Eq. 1). As each activity model is independent of each activity model, they can be distributed across multiple devices, only incurring the overhead of transmitting trajectory matrices and likelihood scores.

For a library of activities A, object GMM models O, object weights w, temporal segment intervals T, and object to feature index mapping function F, we define the activity recognition problem as determining the $a \in A$ for a given input trajectory x that solves the following:

$$\arg\max_{a \in A} \frac{1}{|T_a|} \cdot \sum_{t \in T_a} \Big(\sum_{i=0}^{|O|} w_{a,i}^{t}\Big)^{-1} \cdot \sum_{i=0}^{|O|} w_{a,i}^{t} \cdot \big(\ln \overline{L(x[t, F_{O_i}] | O_{a,i}^{t})}\big) \tag{1}$$

L is the standard GMM likelihood function given an input trajectory x, object to feature index mapping function F, and object model o with GMM component count o_c, means μ_o, and covariance matrices Σ_o, defined as follows:

$$L(x|o) = \sum_{i}^{o_c} w_{o,i} \frac{\exp\left\{\frac{-1}{2} \cdot (x - \mu_{o,i})^{\intercal} \cdot \Sigma_{o,i}^{-1} \cdot (x - \mu_{o,i})\right\}}{(2\pi)^{|F_o|/2} \cdot |\Sigma_{o,i}|^{1/2}} \tag{2}$$

B. Activity Model Training

RAPTOR activity models are trained with a collection of labeled trajectory examples. Each individual activity model is divided into temporal segments, requiring the specification of width and stride parameters that govern the coverage and spacing of each segment (as a percentage of the overall trajectory), respectively. Hyper-parameter optimization techniques or prior knowledge from a subject matter expert could be used to tune these parameters for each activity to improve classification results. Within each temporal segment, a GMM for each object is fit using Expectation-Maximization over the relevant subset of each training trajectory matrix: the rows correspond to that temporal segment (e.g., the first *width%* of every training example) and columns correspond to that object's features (e.g., the X,Y,Z position of a human's torso).

Once object models have been trained for each temporal segment, our algorithm performs feature selection as described in Section III-C. This process identifies the most informative object models within each segment and weights them relative to one another. In order to overcome temporal variations not encountered in training, we modify Eq. 1 with a method inspired by pooling techniques in the Neural Network literature [17], which we describe further in Section III-D.

A key design criteria of online activity recognition is the ability to rapidly classify input trajectories. As the computation time of our classifier scales as a function of activity class count, temporal window count, and object count, it may become computationally infeasible to evaluate all relevant models at the same frequency as the sensor inputs using a single device. In this case, we propose obtaining a lower-fidelity likelihood estimate by fitting a single GMM to the combined features of all relevant objects at each segment and only maintaining this single GMM per segment at test time. This risks overfitting the activity by improperly correlating features, but also reduces the number of models evaluated.

C. Feature Selection

As we are taking an object oriented approach in order to avoid learning false correlations between features, care must be taken to avoid erring too far toward assuming naïve independence between all features. RAPTOR utilizes a two phase feature selection process in an effort to isolate the most informative objects for classifying each activity. The first phase of this process involves computing two new sets of objects. The first object set is generated from an extended feature vector consisting of pairwise object distances computed over the input data. The second set is created from pairwise combinations of objects originally defined in the mapping function F, introducing a total of $|O|^2 - |O|$ new objects to model at each temporal segment per activity model. The second phase of this process involves paring down the number of object models used in each temporal segment, such that only the most informative subset is used.

For each temporal segment of each activity model, a two-class Random Forest Classifier (RFC) [27] is used to discriminate between examples of the target class $\{1\}$ and all other activity classes $\{\text{-}1\}$ using a feature vector comprised of the likelihood scores output from each object model. Therefore, each RFC is trained to perform the function $f : \mathbb{R}^{|O|} \rightarrow \{-1, 1\}$, using the relevant temporal window from each training trajectory across all available activity classes. As there will be substantially fewer instances of the target class, it is important to employ class weighting such that correct classifications of the underrepresented target class are prioritized over correct classifications of non-target class samples during training. The top $\sqrt{|O|}$ features are kept from the RFC at each temporal segment (corresponding to the most informative objects), weighting them proportionately to their importance such that at each segment t, $\sum_{i=0}^{|O|} w_{a,i}^{t} = 1$.

D. Sample Time Variance

Temporal scaling represents a challenging problem for activity recognition: classifiers must be resilient to delays and natural differences in pace between agents while simultaneously taking care not to discard pose ordering or relevant velocity information. One approach that has been used to overcome translational variance in image recognition is max-pooling [17], a process in which a sliding window passes over regions of an image, outputting only the largest value found within that window. As temporal variance in time-series data can be framed as analogous to translational variance in static images, we introduce a modification of max-pooling to strengthen our activity classifiers' resilience to temporal variations, modifying the initial classification equation as follows:

$$\begin{aligned} \arg\max_{a \in A} \frac{1}{|T_a|} \sum_{t \in T_a} \max_{\hat{t} \in [t - \frac{p}{2}, t + \frac{p}{2}]} \Bigg((\sum_{i=0}^{|O|} w_{a,i}^{\hat{t}})^{-1} \\ \cdot \sum_{i=0}^{|O|} w_{a,i}^{\hat{t}} \cdot \left(\ln \overline{L(x[t, F_{O_i}] | O_{a,i}^{\hat{t}})} \right) \Bigg) \end{aligned} \tag{3}$$

By introducing the $\max$ operator over a temporal window parameterized by p, each segment t of the input trajectory x is scored according to the best-fitting segment model in the p-neighborhood of t. This operator forgives a limited but configurable amount of temporal misalignment while still allowing for an implicit approach to pose modeling accomplished by the time-segmented ensemble architecture. With this modification, RAPTOR provides two mechanisms to robustly guard against misclassification as a result of temporal disparities between training and test data: temporal

segment (width and stride) and max-pooling (width) parameters that can be specified for each activity.

As a preliminary step for handling partial trajectories, RAPTOR segments all input trajectories shorter than the mean training trajectory length as if they were the mean length, in order to avoid improper temporal segmentation. While this may result in temporal segments receiving no data to evaluate, classification accuracy is not affected as it is accounted for in the calculation of $|T_a|$ in Equations 1 and 3.

E. Activity Classification

Classification is performed by solving for a in Equation 3, following the pipeline depicted in Figure 1. The provided input trajectory is passed into each activity classifier in order to compute its likelihood under the prior imposed by each particular model. Within each classifier, the input trajectory is partitioned into windows fitting the width and position of each temporal segment. The input signal is further partitioned by object within each temporal segment for each positively weighted object GMM ($\vec{w}_a^t$ in Equations 1 and 3). The entire timespan within the temporal segment is evaluated within each object model, outputting the mean log likelihood across the input frames (L in Equations 1 and 3).

F. Anomaly Detection and Explanation

RAPTOR's object-oriented ensemble approach can be leveraged to provide interpretable explanations for anomalies in input trajectories, as recognized from the perspective of each activity model. Here, we introduce one such method that incorporates outlier detection within and across temporal segments in order to identify portions of input trajectories that fit learned activity models particularly well (or poorly).

In human-robot collaborative scenarios, it is important to synchronize expectations amongst teammates and to be able to understand discrepancies between those expectations and reality. Consider a support robot designed to take items from human workers and carry them across a factory floor in order to reduce the workers' physical stress: if this support robot has difficulty distinguishing between when a human is carrying something versus walking unencumbered, it could be considered unreliable and would likely be removed from the workflow. It would be far easier to diagnose, understand, and debug misclassifications between 'carry' and 'walk' behaviors if the human could ask the robot to explain why it misclassified one as the other (e.g., 'carry' as 'walk') and receive a response from the 'carry' model akin to "In the middle and end of the trajectory, the left hand and right hand features were very poorly matched to my template for 'carry.'"

Template-based approaches have been shown to be effective within similar human-robot interaction settings [28]. By identifying outliers using Algorithm 1, we can employ a template-based approach to summarize the results in an interpretable manner that is similar to the above example. To abstract the granularity of the summaries from the individual activity model parameters, we introduce a separate level of trajectory division for the summarization step, allowing for intuitively labeled divisions (such as ['beginning', 'middle', 'end'] or ['first half', 'second half']) rather than *temporal segment 3*. We additionally define labels to communicate within-division coverage, allowing for responses conveying information about the specificity of the explanation, such as ['some', 'most'] corresponding to explanations that cover up to 50% and over 50% of the time division being described, respectively. This allows the classifier to express more specific information, such as "In *some of* the *first half* of the trajectory, the head features were exceptionally well matched to my model compared with the others."

To describe temporal segment outliers, we use the summary template, "In general the features I was tracking for {coverage} of the {division} were very [well matched / unfamiliar according] to my model for {activity}." For summarizing object outliers within time segments, we use the summary template "In the {division} of the action, the {object list} features were [extremely well matched to my model / did not match my model at all] for {activity}."

This feature is helpful in two important use cases within human-robot interaction: 1) providing insights into a robot collaborator's activity models for improved mutual understanding and expectation setting; and 2) in active learning scenarios, assisting with the coaching of an interaction partner to provide more helpful demonstrations for a learning robot.

Algorithm 1: Find Temporal and Feature-based Outliers

Input: Activity classifier a, Temporal segments list T_a, Object list O, outlier threshold α, Input trajectory x

Output: Lists of temporal segment and segment-indexed object outliers by likelihood

$\vec{l} \leftarrow |O|\times$1-sized vector of weighted object likelihood scores of x averaged across all temporal segments from a;

$traj_m \leftarrow mean(\vec{l}), traj_\sigma \leftarrow stdev(\vec{l})$;

$\vec{ts}_{means} \leftarrow |T_a|\times$1-sized vector of weighted object likelihoods of x averaged within each temporal segment of a;

$\vec{ts}_{outliers} \leftarrow \{ts \in \vec{ts}_{means} \mid \|ts - traj_m\|_1 \geq \alpha \cdot traj_\sigma\}$;

$\vec{obj}_{scores} \leftarrow [|T_a|\times|O|]$ matrix of object likelihood scores of x at every temporal segment;

$\vec{obj}_{outliers} \leftarrow \{\}$;

foreach $t \in T_a$ **do**

  $t_m \leftarrow mean(\vec{obj}_{scores}[t, :])$;

  $t_\sigma \leftarrow stdev(\vec{obj}_{scores}[t, :])$;

  $\vec{obj}_{outliers}[t] \leftarrow \{o \in O \mid \|\vec{obj}_{scores}[t, idx(o)] - t_m\|_1 \geq \alpha \cdot t_\sigma\}$;

return $\vec{ts}_{outliers}$, $\vec{obj}_{outliers}$;


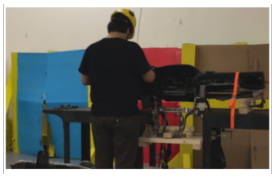


Fig. 2: Images from the UTKinect dataset (Left), Dynamic-AutoFA assembly task (Middle), and Static-Reach task (Right).

IV. Evaluation and Results

We evaluate RAPTOR using three activity datasets: a publicly available single person activity dataset with Kinect-derived skeletal position features (UTKinect [11]); a motion-capture dataset of reaching behaviors from a stationary manufacturing task (Static-Reach [7]); and a new motion capture dataset of a mobile automotive final assembly manufacturing task (Dynamic-AutoFA) depicted in Figure 2.

The UTKinect dataset consists of 10 activity classes performed twice each by 10 individuals: pull, push, walk, carry, wave hands, clap hands, stand up, sit down, and throw. Each activity contains a time-series of 60 features, corresponding to the X,Y,Z coordinates of 20 skeletal joints as estimated from RGB+D video data. Classifications within UTKinect are made more challenging by the presence of occlusions, varying demonstration lengths, and substantial variation between subjects during each activity.

The Static-Reach dataset consists of 16 action classes recorded during a collaborative manufacturing task performed by a human-robot team: forward reaches to place bolts in eight different locations, and their eight corresponding arm-retraction motions. Each data frame consists of three features marking the human's hand position as captured by a PhaseSpace motion capture system. In this dataset, 20 participants performed each action twice over the course of completing two separate, full task executions. As the robot partner utilized different motion planning strategies during each task execution, there was variation between the instances of reaching behaviors due to the human accommodating (avoiding) the robot's motion.

The Dynamic-AutoFA dataset consists of between 15 and 45 instances each of 11 action classes comprising an automotive final assembly task: 'Scan Dashboard,' 'Move to Dashboard,' 'Move to Parts Table,' 'Scan Speedometer,' 'Get Speedometer,' 'Install Speedometer,' 'Get Nav Unit,' 'Scan Nav Unit,' 'Install Nav Unit,' 'Get Scanner,' and 'Place Scanner.' Activity segments consist of 21 features mapping to the X,Y,Z coordinates of the position of the human worker's head, left hand, and right hand, as well as the positions of the parts table, navigation unit, speedometer, and dashboard, as captured by a VICON motion capture system. Classifications within Dynamic-AutoFA are made more challenging through sensor occlusion and noise, as well as substantially overlapping activity classes (such as 'Get Nav Unit' and 'Scan Nav Unit', which were often performed concurrently).

To evaluate RAPTOR using the UTKinect and Static-Reach datasets, we performed 10-fold cross-validation across subjects, training the RAPTOR activity models on demonstrations from all subjects but one and testing against all demonstrations from the held-out actor. For the Dynamic-AutoFA dataset, we performed 10-fold cross-validation within activities by training on 90% of the available demonstrations for each activity and testing on the remaining 10%. We fixed the temporal segment parameters at 12.5% width and 6.25% stride for each activity, yielding 15 temporal segments per activity model. The max-pooling width parameter (p in Equation 3) was fixed at 4. Our results indicate that RAPTOR achieved real-time performance and state-of-the-art accuracy both for activity recognition (classifying full trajectories) and activity prediction (identifying actions mid-execution, given a partial trajectory) without any preprocessing of the input data (e.g., smoothing, alignment, or filtering).

Real-time UTKinect Activity Recognition Accuracy	
Classifier	**Accuracy**
Slama et al. (2015) [21]	88.5%
Chrungoo et al. (2014) [18]	89.45%
Xia et al. (2012) [11]	90.9%
Wang et al. (2015) [23]	90.9%
Devanne et al. (2013) [20]	91.5%
RAPTOR (proposed method)	**92.1%**

TABLE I: Top-1 activity classification accuracies during cross-subject evaluation for the UTKinect dataset for recognition methods self-identified as 'real-time'.

A. Activity Recognition and Prediction

Real-time activity recognition is critical for task planning for collaborative, multi-agent scenarios. In Table I we report best-in-class performance by RAPTOR compared with real-time methods found in the literature. We report recognition accuracy across all three datasets for both Top-1 and Top-2 classification thresholds (where the top two predictions for each trajectory are considered) in Table II.

Our method demonstrates extremely high-level performance on the manufacturing task datasets while achieving the highest accuracy for real-time activity recognition on UTKinect. We include Top-2 accuracy results as many sequential tasks incorporate task priors that can be used to bias the final activity likelihoods generated as output by our approach. As a representative example, proper task procedure for the automotive final assembly task captured in the Dynamic-AutoFA dataset dictates that the installation of the speedometer must occur before installation of the navigation unit due to the component wiring layout. Accordingly, a task-level prior would strongly penalize the likelihood of the 'Install Nav Unit' activity occurring before the 'Install Speedometer' activity was completed.

RAPTOR Activity Recognition Accuracy		
Dataset	**Top-1 Accuracy**	**Top-2 Accuracy**
UTKinect	92.1%	99.4%
Static-Reach	97.5%	99.4%
Dynamic-AutoFA	92.0%	100.0%

TABLE II: Classification accuracy scores for RAPTOR across a general activity dataset (UTKinect) and two domain-specific manufacturing task datasets. The leading real-time human reaching predictor [24] achieves 91.9% Top-1 and 97.8% Top-2 accuracy on Static-Reach.

RAPTOR Online Activity Prediction Accuracy				
Dataset	**25%**	**50%**	**75%**	**100%**
UTKinect	79.4%	83.1%	84.7%	92.1%
Static-Reach	69.7%	77.2%	93.8%	97.5%
Dynamic-AutoFA	91.7%	88.1%	90.5%	92.0%

TABLE III: Top-1 classification accuracy scores for varying partial trajectory lengths on each of the three datasets.

B. Online Activity Prediction

RAPTOR also performs well with partial trajectory classification, capable of real-time prediction at a frequency comparable to the sensor capture rate (30Hz) using an unoptimized Python implementation on commodity hardware (Intel i7-2600K CPU). While the computation time of each activity model's likelihood scores for a given trajectory will change as a function of trajectory length (frame count) and the model's parameters (temporal segment width, temporal segment stride, and max-pooling width), our experiments resulted in a mean classification time (on full-length trajectories) per model of 36.7ms on UTKinect, 38.5ms on Static-Reach, and 38.8ms on Dynamic-AutoFA. These times could be further reduced by parallelizing the computation of temporal segment likelihood scores for each activity model.

Table III depicts Top-1 partial trajectory classification accuracy for all three datasets, including results from when our classifier is given 25%, 50%, 75%, and 100% of each test trajectory. At 25%, the activity classifier is often using less than 500ms of motion data per activity; as such, we observed classification errors across activities with similar initial movements. In the Dynamic-AutoFA dataset, 'Scan Dashboard' and 'Move to Dashboard' were often confused as they tended to have partially overlapping annotations in the training set. Likewise, 'Install Nav Unit' and 'Get Scanner' were often confused as both actions tended to require reaching towards the same location (workers often placed the Scanner tool on top of the dashboard assembly when not using it). Within the UTKinect dataset, we observed confusion amongst 'pull' and 'throw,' as well as between 'pickup' and 'sit down,' at 25% of the partial trajectory due to initial motion similarities.

C. Anomaly Detection and Explanation

In order to evaluate the interpretable aspect of our classifier, we demonstrate its ability to use anomaly explanation to clarify classification errors that would otherwise be difficult to diagnose. Given the confusion matrix from our UTKinect activity recognition evaluation (Figure 3), it may be useful to understand why 'push' was misclassified as 'throw' or why 'carry' was occasionally misclassified as 'walk.' Here, using the approach described in Section III-F, we detail how to generate an explanation to understand and debug these misclassifications, with the observed 'carry'/'walk' confusion serving as a representative example.

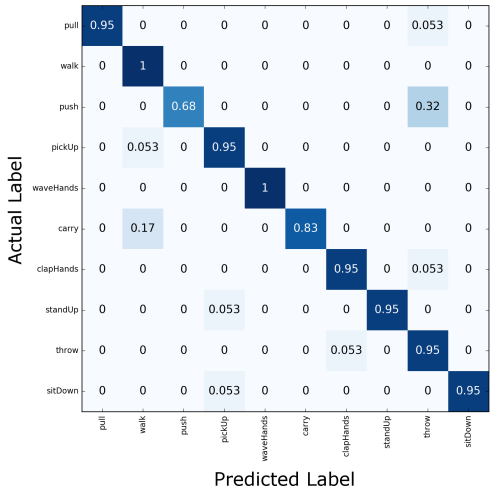

Fig. 3: Confusion Matrix for UTKinect dataset recognition (100% trajectory) results with time segment width 12.5%, stride 6.25% and max-pooling parameter $p = 4$. The confusion between 'push' and 'throw' suggests the max-pool parameter could be reduced for the 'push' activity model.

For our experiments, we defined three trajectory divisions: 'beginning,' 'middle,' and 'end,' corresponding to the first 33%, middle 34%, and final 33% of the input trajectory, respectively. To summarize the frequency with which an object occurs as an outlier within these divisions, we defined four coverage categories: 'a bit,' 'some,' 'most,' and 'all' – corresponding to the intervals $[0, 25\%), [25\%, 50\%], (50\%, 100\%)$, and $[100\%]$, respectively. We set $\alpha = 2.0$ in Algorithm 1.

When queried with a misclassified instance of 'carry' from the UTKinect dataset, the RAPTOR activity model of 'carry' provided the following explanation: "In the beginning of the action, right hand and right wrist features did not match my model for 'carry' at all. In the middle of the action, right hand, right wrist, and left wrist features did not match my model for 'carry' at all. In the end of the action, left hand and right hand features did not match my model for 'carry' at all." With this information, it is clear that either the hand model for 'carry' is too restrictive or this particular example has a unique motion hand pattern ($\geq 2\sigma$ likelihood deviation) that has not been previously observed. This identifies the features attributable to the classification error, allowing targeted training examples to be provided in order to repair the model.

When applied to properly classified trajectories, our approach can identify areas of weakness or strength in the

activity model even if they were not enough to generate classification errors. Within the Dynamic-AutoFA dataset, a poorly executed 'Place Scanner' activity that was still classified correctly output the explanation: "In general, the objects I was tracking for some of the middle of the action were especially unfamiliar according to my model for 'Place Scanner."' In this case, the scanner tool was dropped and the overall likelihood of the temporal segment during the recovery of the tool was substantially lower than other segments for that trajectory, illustrating a successful anomaly detection and summarization.

V. Conclusion

We have presented Rapid Activity Prediction Through Object-oriented Regression (RAPTOR), a highly parallel activity classifier capable of performing real-time activity recognition given only partial trajectories, capable of achieving state-of-the-art accuracy on three diverse activity datasets, even in the absence of data preprocessing and hyperparameter tuning. We described and demonstrated our approach's ability to perform anomaly detection and explanation, autonomously providing interpretable summaries to improve collaborators' understanding of the learned activity models. Our contribution, targeted toward improving human-robot collaboration, improves upon prior work in terms of overall classification accuracy, online performance, and interpretability.

References

[1] C.-M. Huang and B. Mutlu, "Anticipatory robot control for efficient human-robot collaboration," in *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 2016, pp. 83–90.

[2] B. Hayes and B. Scassellati, "Effective robot teammate behaviors for supporting sequential manipulation tasks," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2015)*, 2015.

[3] J. Mainprice and D. Berenson, "Human-robot collaborative manipulation planning using early prediction of human motion," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, 2013, pp. 299–306.

[4] K. Hausman, S. Niekum, S. Osentoski, and G. S. Sukhatme, "Active articulation model estimation through interactive perception," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 3305–3312.

[5] T. Lan, L. Sigal, and G. Mori, "Social roles in hierarchical models for human activity recognition," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1354–1361.

[6] B. Hayes and B. Scassellati, "Autonomously constructing hierarchical task networks for planning and human-robot collaboration," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2016.

[7] P. A. Lasota and J. A. Shah, "Analyzing the effects of human-aware motion planning on close-proximity human–robot collaboration," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 57, no. 1, pp. 21–33, 2015.

[8] P. Koniusz, A. Cherian, and F. Porikli, "Tensor representations via kernel linearization for action recognition from 3d skeletons," *arXiv preprint arXiv:1604.00239*, 2016.

[9] I. Gori, J. Aggarwal, L. Matthies, and M. Ryoo, "Multitype activity recognition in robot-centric scenarios," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 593–600, 2016.

[10] E. Cippitelli, S. Gasparrini, E. Gambi, and S. Spinsante, "A human activity recognition system using skeleton data from rgbd sensors," *Computational intelligence and neuroscience*, vol. 2016, 2016.

[11] L. Xia, C. Chen, and J. Aggarwal, "View invariant human action recognition using histograms of 3d joints," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*. IEEE, 2012, pp. 20–27.

[12] J. Burke and R. Murphy, "Human-robot interaction in usar technical search: Two heads are better than one," in *Robot and Human Interactive Communication, 2004. ROMAN 2004. 13th IEEE International Workshop on*. IEEE, 2004, pp. 307–312.

[13] M. Johnson, J. M. Bradshaw, P. J. Feltovich, C. M. Jonker, B. Van Riemsdijk, and M. Sierhuis, "The fundamental principle of coactive design: Interdependence must shape autonomy," in *Coordination, organizations, institutions, and norms in agent systems VI*. Springer, 2011, pp. 172–191.

[14] B. Hayes and B. Scassellati, "Challenges in shared-environment human-robot collaboration," in *the "Collaborative Manipulation" Workshop at the 8th ACM/IEEE International Conference on Human-Robot Interaction*, 2013.

[15] S. Nikolaidis and J. Shah, "Human-robot cross-training: Computational formulation, modeling and evaluation of a human team training strategy," in *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction*. IEEE Press, 2013, pp. 33–40.

[16] J. Baraglia, M. Cakmak, Y. Nagai, R. Rao, and M. Asada, "Initiative in robot assistance during collaborative task execution," in *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 2016, pp. 67–74.

[17] D. Scherer, A. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," in *International Conference on Artificial Neural Networks*. Springer, 2010, pp. 92–101.

[18] A. Chrungoo, S. Manimaran, and B. Ravindran, "Activity recognition for natural human robot interaction," in *International Conference on Social Robotics*. Springer, 2014, pp. 84–94.

[19] S. Gaglio, G. L. Re, and M. Morana, "Human activity recognition process using 3-d posture data," *IEEE Transactions on Human-Machine Systems*, vol. 45, no. 5, pp. 586–597, 2015.

[20] M. Devanne, H. Wannous, S. Berretti, P. Pala, M. Daoudi, and A. Del Bimbo, "Space-time pose representation for 3d human action recognition," in *International Conference on Image Analysis and Processing*. Springer, 2013, pp. 456–464.

[21] R. Slama, H. Wannous, M. Daoudi, and A. Srivastava, "Accurate 3d action recognition using learning on the grassmann manifold," *Pattern Recognition*, vol. 48, no. 2, pp. 556–567, 2015.

[22] L. L. Presti, M. La Cascia, S. Sclaroff, and O. Camps, "Gesture modeling by hanklet-based hidden markov model," in *Asian Conference on Computer Vision*. Springer, 2014, pp. 529–546.

[23] P. Wang, W. Li, Z. Gao, J. Zhang, C. Tang, and P. Ogunbona, "Deep convolutional neural networks for action recognition using depth map sequences," *arXiv preprint arXiv:1501.04686*, 2015.

[24] C. Pérez-D'Arpino and J. A. Shah, "Fast target prediction of human reaching motion for cooperative human-robot manipulation tasks using time series classification," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[25] S. Alexandrova, M. Cakmak, K. Hsiao, and L. Takayama, "Robot programming by demonstration with interactive action visualizations," *Proceedings of Robotics: Science and Systems, Berkeley, USA*, 2014.

[26] C. Diuk, A. Cohen, and M. L. Littman, "An object-oriented representation for efficient reinforcement learning," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 240–247.

[27] T. K. Ho, "Random decision forests," in *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, vol. 1. IEEE, 1995, pp. 278–282.

[28] M. Cakmak and A. L. Thomaz, "Designing robot learners that ask good questions," in *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*. ACM, 2012, pp. 17–24.