Beyond Action Recognition: Action Completion in RGB-D Data

Farnoosh Heidarivincheh
farnoosh.heidarivincheh@bristol.ac.uk
Majid Mirmehdi
majid@cs.bris.ac.uk
Dima Damen
dima.damen@bristol.ac.uk

Computer Science Department
University of Bristol
Bristol, UK

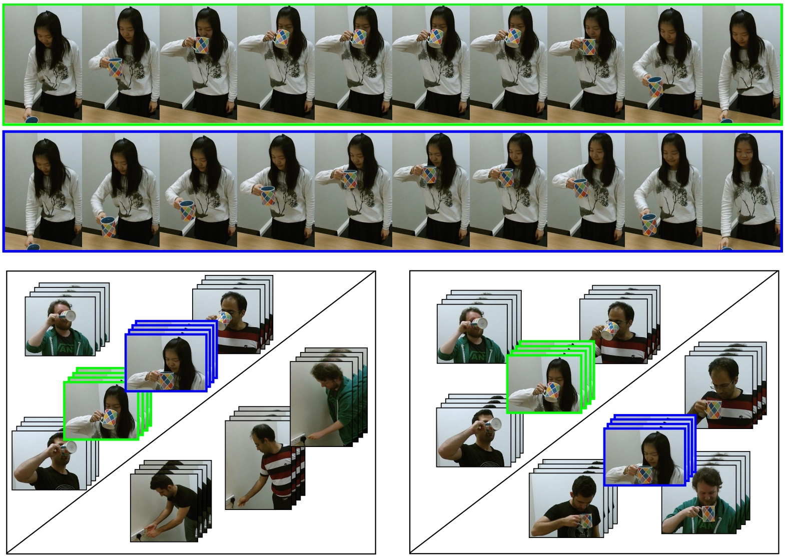
Abstract

An action is *completed* when its goal has been successfully achieved. Using current state-of-the-art depth features, designed primarily for action recognition, an incomplete sequence may still be classified as its complete counterpart due to the overlap in evidence. In this work we show that while features can perform comparably for action recognition, they vary in their ability to recognise incompletion. Experimenting on a novel dataset of 414 complete/incomplete object interaction sequences, spanning six actions and captured using an RGB-D camera, we test for completion using binary classification on labelled data. Results show that by selecting the suitable feature per action, we achieve 95.7% accuracy for recognising action completion.

1 Introduction

Robust motion representations for action recognition have achieved remarkable performance in both controlled and 'in-the-wild' scenarios. Such representations are primarily assessed for their ability to label a sequence according to some predefined action classes (e.g. *walk, wave, open*). Although increasingly accurate, these classifiers are likely to label a sequence, even if the action has not been fully completed, because the motion observed is similar enough to the training set. Consider the case where one attempts to drink but realises the beverage is too hot. A *drinking-vs-all* classifier is likely to recognise this action as *drinking* regardless. We introduce the term **action completion**, which aims to recognise whether the action's goal has been successfully achieved. This is conceptually different from, but very related to, action recognition. In other words, in addition to attempting to assign a class label to an observed video, we want to confirm whether the person has completed a known action.

The notion of completion differs per action. For *drinking*, the action is *completed* when one actually consumes a beverage from a cup. Alternatively, for *filling*, the action is *completed* when the container becomes full. While for some actions, it is either infeasible or too dependent on the viewing angle to verify completion using a visual sensor (e.g. *talking* or *reading*), in many actions, including the examples above, an observer would be able to make the distinction by noticing subtle differences in motion.



(a) *drink* vs. *plug* (b) complete vs. incomplete (of *drink*)

Figure 1: For a complete *drink* (green) and an incomplete *drink* (blue) sequences from our dataset, both are classified as *drink* when using *drink* vs. *plug* classifier (a). The proposed supervised action completion model (b) identifies the incomplete sequence.

Incompletion could result from negligence or forgetfulness, or could be deliberate as one only pretends to complete an action. Incompletion could also be a result of difficulties in performing the action despite a genuine attempt, *e.g.* hitting the golf ball into the hole. Applications for recognising incompletion thus span healthcare, surveillance, and automatic training, amongst others. In this work, we focus on object interactions, i.e. the subset of actions where a person interacts with one or more objects in their environment (i.e. *open, drink, pull*). We test and report results using RGB-D data, however, the action completion argument presented here could be applied to RGB data, as well as other actions. Our focus is motivated by the application of Smart Homes, for example as in the SPHERE project [12], where visual sensing can help determine, for example, whether an elderly person with dementia has actually taken their medicine or have closed the tap.

We address incompletion in a supervised approach, using a dataset that contains complete as well as incomplete sequences, spanning 6 actions (*switch, plug, open, pull, pick* and *drink*). We investigate the ability of state-of-the-art depth features, initially designed for action recognition, to distinguish completion of actions. Results show that the performance of these features varies for recognising completion per action class. We then propose a general model for action completion that uses cross-validation on the training set to select the best features for assessing action completion per action. The overall concept of the action completion problem and our proposed model are illustrated in Figure 1.

The remainder of this paper is organised as follows: related works in Sec. 2, the method and features used in Sec. 3, a new dataset of 414 complete and incomplete sequences in Sec. 4, results in Sec. 5, and finally conclusion and future work in Sec. 6.

2 Related Work

To the best of our knowledge, no previous work has attempted *action completion* in RGB or RGB-D data. We know of only the works of Soran *et al.* who have considered predicting missing actions within an activity in RGB data [6] and Wang *et al.* who recognise complete actions by studying the effect of the action on an environment [11]. We first review works on action recognition in RGB-D, and then reflect on [6, 11] and their relationship to our work.

Action Recognition in RGB-D data - Many methods for action recognition using RGB-D data rely on skeletal joints as extracted from Kinect SDK or OpenNI [1, 3, 7, 14, 15]. In [3], an action is represented as a sequence of the most informative joints. Sequences are partitioned into temporal segments and the means and variances of joint angles and the maximum angular velocity of joints are calculated and rank-ordered over these segments. Then SVM and KNN are used for classification. In [14], spatial histograms of joints locations, defined in a spherical coordinate system, are clustered into posture visual words. Dynamics are modelled using an HMM. In [1], the relative position of joint quadruples are proposed as a new feature. These are encoded using Fisher vectors and classified using a linear SVM. In [7], joint positions are combined with motion, hand position and appearance features, before using a hierarchical Maximum Entropy Markov Model to represent the action. In [15], the Eigenjoints feature is proposed as the difference in joint positions within and across frames. Discriminative features are then selected and a KNN classifier is used.

Some works have used raw depth data directly [8, 9, 13, 16]. In [16], HOG features are computed from depth motion maps, projected onto three orthogonal Cartesian planes. Actions are classified using a linear SVM. In [9], sampled sub-volumes from temporal depth data are selected as their most discriminative feature which is robust to occlusion by modelling noise as the reconstruction error of sparse coding. In [8], depth maps are partitioned into 4D cells along space and time axes. Then, the occupancy information in these spatio-temporal cells is used as a feature. In [4], Histogram of Oriented 4D Normal Vectors (HON4D) descriptors are proposed as histograms of the surface normals from depth map sequences and the discriminative features are passed to an SVM for classification. In [13], spatio-temporal interest points are extracted from depth data and represented using information from the 3D cuboids around the interest points. The features are encoded using bag-of-words before classification by an SVM.

A novel encoding of both joint and depth features, using short-time Fourier transform, is proposed in [10]. This encoding, combined with actionlet ensemble modelling, achieves robust performance for recognising a variety of daily actions, including object interactions. Combining joint positions with depth data around these joints, referred to as local occupancy patterns, is particularly suitable for capturing the relationships between body parts and environmental objects [2]. In this work, we use the encoding from [10] as it suits our dataset of object interactions.

Action and Activity Completion in RGB data - Two recent works have attempted to detect missing actions [6] or model the effect of an action on the environment [11] - making them the closest works to the *action completion* problem we introduce here.

In [11], an action is defined as a transformation from some starting state before the action begins, called precondition state, to the state related to some end frames after the action is completed, called effect. This transformation, learnt from training data using CNNs is used for action recognition and is tested on several RGB datasets. While this approach could be used for detecting completion, in this work we focus on the motion itself, rather than the start and end states solely. The closest work to ours is [6] which attempts to detect missing sub-activities from a sequence representing an activity (making latte), modelled as a flexible ordered graph. Even if we consider that these missing parts express a kind of incompleteness on the activity level, we differ from this approach in two ways. First, we aim to detect incompleteness when the action is attempted but not completed (e.g. attempting to drink but not actually drinking). Second, such an approach would require prior knowledge of semantically sensible sub-actions, and is sensitive to the number of sub-actions and their correct labelling.

In summary, in this work we focus on *action completion* as opposed to higher level activities or sequences of actions. We assume the action has been attempted and focus on detecting completion. In contrast to [11], we study the observed motion rather than the effect of the action on the surrounding environment.

3 Proposed Method for Recognising Action Completion

We now propose a supervised approach for action completion that relies on labelled complete and incomplete samples. Since the notion of completion differs per action, a general action completion method should investigate the performance of different types of features to accommodate the various action classes. For example, for the action *pick*, the difference between complete and incomplete actions originates from the subtle change in body pose when holding an object, or by observing an object in the hand. In contrast, for the action *drink*, the speed at which the action is performed is better able to assess the completion. In Section 3.1, we review a number of state-of-the-art depth action recognition features. We then propose a method that attempts to choose the feature(s) suitable for recognising completion from the pool of depth features. The method is based on cross-validation over labelled training data and is explained in Section 3.2.

3.1 RGB-D Data and Feature Extraction

Given a video sequence of an action being performed, captured using an RGB-D sensor, we first extract skeleton data from every frame of the sequence using Kinect for Windows SDK 2.0 which estimates joint positions using the method from [5]. For each frame, 16 joint positions are estimated that represent the upper body of the person, as all actions tested in this work relate to object interactions by hand. Noise is smoothed by applying a 1D Gaussian filter to each joint position across time.

As noted earlier, the proposed method expects a pool of features, and assesses the ability of each feature to identify completion for the action modelled, given labelled training data. In this investigation, five features are extracted from skeleton data, previously introduced or used by other works [10, 14, 15, 17]. We select these features in particular as they capture and encode the temporal dynamics of an action:

- Local Occupancy Pattern (LOP): This feature, first introduced in [10], is useful for actions that include human-object interaction. LOP is computed by partitioning the neighbourhood around each joint into cells and counting the number of depth points present in each cell from the point cloud data. These numbers not only show the presence of an object near a joint, but also approximate the shape of the object via spatial binning. The size of our LOP feature is 16×64 per frame.

- Joint Positions (JP): This feature is the 3D coordinates of joints, relative to the *SpineMid* joint. The size of the JP feature is 16×3 per frame.
- Joint Relative Positions (JRP): This feature is the difference between the 3D positions of every pair of joints in the same frame, and its size is 120×3 per frame.
- Joint Relative Angles (JRA): This feature is the 3D vector representing the rotation between each pair of connected joints. Connected joints are those that are connected by a segment to represent the stick figure of a person. Its size is 15×4 per frame.
- Joint Velocities (JV): This feature is the 3D vector representing the displacement of each joint position in consecutive frames, and has a size of 16×3 per frame.

The latter four features use skeletal joints data, while LOP combines joints with depth data. The encoding of the temporal dynamics of an action encapsulated by these features will help us in detecting incomplete actions. Different methods have been suggested for encoding temporal dynamics, such as spatio-temporal pyramids [8] and HMM [14]. In this work, we use the Fourier temporal pyramid, introduced by [10]. In [10], Fourier transform is applied across the whole sequence as the first level of the temporal pyramid. Then, to create further levels of the pyramid, the action is recursively partitioned into segments temporally and short-time Fourier transform is applied to every segment. Using low frequency coefficients of the Fourier transform not only smooths the noise, but also is a good representation of the action dynamics and yields a fixed size feature vector. The features obtained from different levels of the pyramid are concatenated before being passed to the classifier.

3.2 Selecting Features for Action Completion

Given labelled *complete* and *incomplete* sequences of the same action, we build a model of completion of that action as a binary classifier for each of our actions. As explained before, the discriminative features, i.e. those able to separate complete from incomplete sequences, differ for various actions. A general model should thus be able to automatically select the features for each action from a pool of features. This requires assessing the ability of the feature to classify *complete* sequences as *complete*, and *incomplete* sequences otherwise.

We propose to evaluate the performance of each feature, from the pool of features, on the training set using 'leave-one-person-out' cross validation. At each fold in the cross validation, all sequences by one person are removed. As people differ in the way they (in)complete an action, the feature suitable for recognising completion per person might differ. We accumulate evidence across the various folds to rank each feature in the pool of features. The total number of correctly classified sequences is recorded per feature. We rank all features by their accuracy, and select the feature (or features) that performs the best during cross-validation on the training set. By cross-validating on the training set, we attempt to test the generality of the feature to unseen individuals rather than overfit training data.

While the model is built per action, it is independent of the action label *per se*. The method only requires labelled *complete* and *incomplete* sequences and would, provided a rich-enough pool of features, builds an *action completion* model for any action. Once the completion model is built for each action, a test sequence can be checked for completion.

4 Dataset

We are not aware of any datasets in the computer vision community that provides both complete and incomplete samples of different actions. As noted earlier, the 2D egocentric dataset

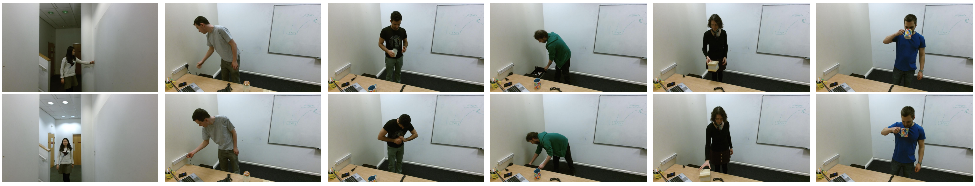


Figure 2: Pairs of complete (top) and incomplete (bottom) sample frames from our dataset of six actions (left to right): *switch, plug, open,pull, pick, drink*

	total #	**# complete**	**# incomplete**	$\mu(sec)$	$\sigma(sec)$
switch	67	35	32	3.87	0.72
plug	73	37	36	8.14	2.74
open	68	36	32	6.83	2.70
pull	71	34	37	6.43	1.70
pick	69	33	36	4.03	1.16
drink	66	34	32	8.83	2.09

Table 1: Dataset specifications: number of sequences, number of complete and incomplete sequences, average (μ) and standard deviation (σ) of sequence lengths per action.

presented in [6], is related to only one activity with its corresponding sub-activities. Thus, we collected a new dataset RGBD-Action-Completion-2016[1] containing 414 sequences using a Microsoft Kinect v2 (see Table 1). The sequences capture six actions, chosen to represent a variety of object interactions: *switch* - turning off a light switch, *plug* - plugging a socket, *open* - opening a jar, *pull* - pulling a drawer, *pick* - picking an item from a desk and *drink* - drinking from a cup. For each action, eight subjects - 5 males and 3 females - performed at least four complete and four incomplete sequences. Sample frames from the dataset are shown in Figure 2. For each action, we varied the conditions so the action cannot be completed as follows:

switch: subjects were asked to pretend they have forgotten to switch the light off,
plug: subjects were given a plug that does not match the socket,
open: a lid was glued to the jar so it could not be opened,
pull: a drawer was locked so could not be pulled,
pick: subjects were asked to attempt to pick an object, and then change their mind,
drink: a mug was filled with very hot water unsuitable for drinking.

5 Experimental Results

In all our results, we test 'leave-one-person-out' cross validation, i.e. all sequences from the one individual are removed before training. The model built is then used to test each sequence from the person 'left-out'. In order to have an overall view on the action completion problem, as well as the proposed method for recognising incompletion, results on four experiments (EA, EB, EC, ED), using the features presented in 3.1, are reported as follows.

(EA) Complete Action Recognition - Comparable to standard RGB-D action recognition works [1, 9, 10, 13, 15, 16], we performed action recognition on the *complete* sequences

[1] From project page: http://www.cs.bris.ac.uk/~damen/ActionCompletion/
or directly at: http://dx.doi.org/10.5523/bris.66qry08cv1fj1eunwxwob3fjz

	LOP	**JP**	**JRP**	**JRA**	**JV**
switch	**100**	99	99	**100**	**100**
plug	**99**	92.3	91.9	92.8	97.1
open	97.6	98.1	**100**	94.7	94.3
pull	**98.1**	91.4	91.4	94.7	92.3
pick	97.6	99.5	**100**	96.7	95.2
drink	99	97.1	98.1	99	**100**
Average	**98.6**	96.3	96.7	96.3	96.5

Table 2: Complete action recognition accuracy: one-vs-all linear SVM for each feature (Experiment EA).

in our proposed dataset. For each action, a one-vs-all linear SVM was trained. Results in Table 2 show the success rate for each feature, demonstrating that all five features produce high % accuracy for action recognition on our dataset, over the variety of tested actions.

(EB) Incomplete Action Recognition - In the second experiment, a binary one-vs-one linear SVM classifier was trained with the *complete* samples of two different actions and tested with their *incomplete* samples. In Table 3, for each pair of actions, we report the % ERROR for classifying an *incomplete* sample of the action as a complete one. For example, the 3rd column for the *switch* action in Table 3 shows that using the LOP feature, all incomplete *switch* samples were indeed classified as *switch*, despite the action being *incomplete*. This is due to the fact that the motion of the *incomplete* action is usually similar to the corresponding *complete* action.

However, we noticed that such confusion depends not only on the action being classified, but also on the feature used. This is an interesting conclusion when compared to Table 2, where all the features obtained comparable and highly accurate results on *complete* sequences. These features, originally designed for action recognition, behave differently on *incomplete* action sequences with only some able to distinguish the subtle changes between *complete* and *incomplete* sequences of an action.

To illustrate this behaviour, we report confusion matrices which present the percentage of an *incomplete* action being classified as another action, for each feature. Again, *complete* samples were used for training, and the classification was performed by finding the nearest neighbour to the *incomplete* test sequence. Figure 3 shows, for example, that when using the LOP feature, incomplete *plug* is 91.9% likely to be classified as complete *plug*, 5.4% as

		LOP	**JP**	**JRP**	**JRA**	**JV**
switch vs.	*plug*	100	100	100	100	100
	open	100	100	93.8	100	87.5
	pull	100	100	100	100	100
	pick	100	90.6	53.1	100	100
	drink	100	100	100	100	96.9

		LOP	**JP**	**JRP**	**JRA**	**JV**
plug vs.	*switch*	100	100	100	100	97.2
	open	100	97.2	100	100	97.2
	pull	97.2	91.7	94.4	88.9	97.2
	pick	100	94.4	97.2	100	100
	drink	97.2	100	100	97.2	69.4

		LOP	**JP**	**JRP**	**JRA**	**JV**
open vs.	*switch*	100	100	100	100	50
	plug	100	100	100	100	25
	pull	100	100	100	100	50
	pick	90.6	100	100	100	100
	drink	100	93.8	100	100	0

		LOP	**JP**	**JRP**	**JRA**	**JV**
pull vs.	*switch*	100	100	100	100	89.2
	plug	64.9	37.8	35.1	46	32.4
	open	100	100	89.2	100	91.9
	pick	100	100	89.2	100	100
	drink	100	97.3	89.2	100	70.3

		LOP	**JP**	**JRP**	**JRA**	**JV**
pick vs.	*switch*	100	100	94.4	94.4	83.3
	plug	100	55.6	55.6	80.6	30.6
	open	100	41.7	50	88.9	47.2
	pull	91.7	50	55.6	88.9	22.2
	drink	80.6	100	100	100	91.7

		LOP	**JP**	**JRP**	**JRA**	**JV**
drink vs.	*switch*	100	81.3	62.5	100	34.4
	plug	100	78.1	65.6	93.8	37.5
	open	90.6	18.8	46.9	84.4	15.6
	pull	100	65.6	43.8	96.9	46.9
	pick	65.6	15.6	6.3	56.3	62.5

Table 3: For each pair of actions, incomplete action recognition results obtained by one-vs-one linear SVM classification across the different features (Experiment EB).

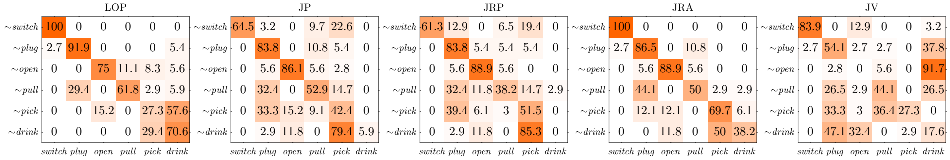


Figure 3: Confusion matrices obtained from *1-NN* classification of incomplete sequences (specified with ∼).

		LOP	**JP**	**JRP**	**JRA**	**JV**
switch	complete	100	94.3	94.3	100	100
	incomplete	100	75	75	100	100
	total	**100**	85.1	85.1	**100**	**100**
plug	complete	91.9	94.6	89.2	83.8	91.9
	incomplete	75	80.6	66.7	75	97.2
	total	83.6	87.7	78.1	79.5	**94.5**
open	complete	94.4	94.4	94.4	91.7	94.4
	incomplete	100	96.9	100	100	100
	total	**97.1**	95.6	**97.1**	95.6	**97.1**
pull	complete	79.4	70.6	73.5	85.3	91.2
	incomplete	94.6	73	81.1	91.9	97.3
	total	87.3	71.8	77.5	88.7	**94.4**
pick	complete	97	93.9	97	97	100
	incomplete	88.9	94.4	100	100	91.7
	total	92.8	94.2	**98.6**	**98.6**	95.7
drink	complete	94.1	94.1	94.1	94.1	100
	incomplete	100	100	100	100	100
	total	97	97	97	97	**100**

Table 4: Complete vs. incomplete action results (Experiment EC). Accuracy is reported for both complete and incomplete sequences, separately, as well as the total for their union.

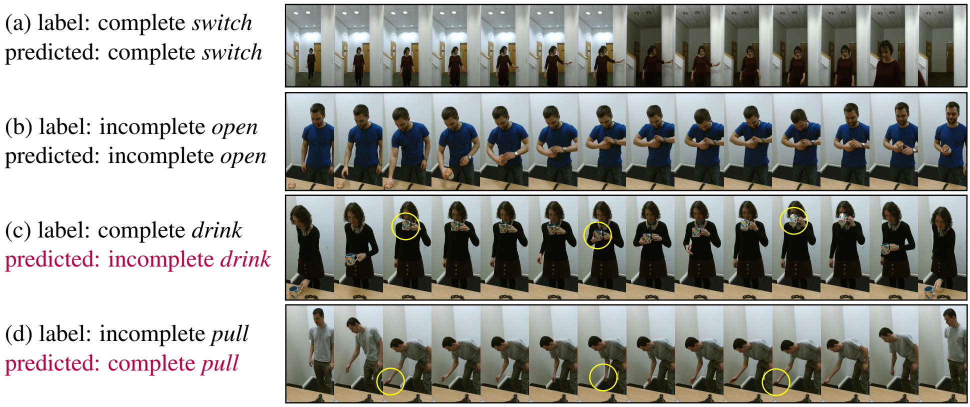
complete *drink* and 2.7% as complete *switch*. These results confirm that the chosen features, originally designed for action recognition, vary in their ability to classify incomplete action sequences. JV for example deviates significantly from the diagonal, showing the sensitivity of the feature to subtle changes resulting potentially from incompletion.

(EC) Complete vs. Incomplete Action Recognition - We then trained a binary linear SVM for complete vs. incomplete sequences of the same action for each feature. Both *complete* and *incomplete* samples of the same action were used in training and testing, without any overlaps. The results in Table 4 again show that the features have different success rates for the various actions. For example, the Joint Velocities (JV) feature significantly outperformed other features for actions *plug* and *pull*, because in these two cases *complete* and *incomplete* sequences differ in the speed at which the actions are performed. On the other hand, for action *pick*, JV did not produce the best results as both incomplete and complete sequences have comparable speeds. Here, JRP and JRA outperform the other features, due to the change in body pose when holding an object.

(ED) Selecting Features for Action Completion - As features vary in their ability to classify *complete* vs. *incomplete* sequences for different actions, a general *action completion*

	Subjects								
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**total**
switch	100	100	100	100	100	100	100	100	100
	LOP, JRA, JV	LOP, JRA, JV	LOP, JV	LOP, JV	LOP, JV	LOP, JRA, JV	LOP, JV	LOP, JV	
plug	83.3	100	87.5	100	88.9	100	100	100	94.5
	JV	JV	JV	JV	JV	JV	JV	JV	
open	100	85.7	100	100	100	87.5	90	100	95.6
	JV	JV	JP, JRP	LOP, JRP, JV	JRP	JRA	JV	LOP, JRP, JRA, JV	
pull	88.9	100	100	100	100	87.5	80	100	94.4
	JV	JV	JV	JRA, JV	JV	JV	JV	JV	
pick	90	100	100	100	100	100	50	100	92.8
	JRA	JRA	JRA, JV	JP, JRA	JRA	JRP, JRA	LOP, JRA	JRA	
drink	77.8	100	100	100	100	100	100	100	97
	LOP, JP, JRP, JRA, JV	JV	JV	JV	JV	JV	JV	JV	
								total	95.7

Table 5: Results for general action completion model (Experiment ED).



(a) label: complete *switch*
predicted: complete *switch*

(b) label: incomplete *open*
predicted: incomplete *open*

(c) label: complete *drink*
predicted: incomplete *drink*

(d) label: incomplete *pull*
predicted: complete *pull*

Figure 4: Sample frames of correctly (a), (b) and incorrectly (c), (d) classified test sequences. In (c), the person hesitates and adjusts her cup before completing a *drink*, making the sequence more similar to an incomplete *drink*. In (d), using JV solely, the hand seems to perform a *pull* in full even when the drawer remains unmoved. Again the motion is similar to a complete *pull*.

model, which is capable of detecting *incomplete* actions, should be able to determine the best feature(s) for that particular action among the pool of features. We performed this automatically by cross validation on training data using the different features separately. The feature with the maximum accuracy on the training data was selected to build the completion model. When multiple features performed equally well, they were concatenated. Table 5 shows the results for the proposed model and presents the accuracy and the chosen feature(s) for each test case, i.e. each 'leave-one-person-out' fold per action. The overall accuracy is reported for all subjects. The results show high success rates compared to the best performance in Table 4, especially for *plug*, *pull*, and *switch* actions.

In most cases in Table 5, the feature(s) producing the highest accuracy was indeed selected and the sequences were correctly classified as either *complete* or *incomplete*. Failure arises when the motion performed is different for the test subject. Examples of success and failure[2] are shown in Figure 4. Across all our *complete* and *incomplete* sequences, actions and subjects, automatic feature selection enables 396 sequences to be correctly classified - that is 95.7% of the sequences in the dataset.

[2] Video results at: http://youtu.be/iBdW-kVKMds

6 Conclusion and Future Work

In this work, we introduced the term *action completion* as a step beyond the task of action recognition where, in application areas such as Healthcare and Surveillance, it is important to ensure the recognised action has indeed been completed. For example, consider the case of an elderly with dementia who lives in a smart home. Specifying if they have fully closed the fridge door or the kitchen tap or actually taken their medicine is critical. We have collected a dataset of *complete* and *incomplete* action sequences which we avail to the vision community. Our experiments showed that while various features from skeleton and depth data perform comparably for the task of action recognition, these features vary in their ability to recognise completion. Moreover, these features have varying performance over different actions. We proposed a method for selecting the best features for recognising completion per action. Tested on various subjects and actions, automatic selection of features produces highly accurate recognition of *complete* and *incomplete* sequences.

For future work, new features as well as a wider variety of action classes, potentially beyond object interactions, should be investigated towards analysing the differences between *complete* and *incomplete* sequences. Pre-trained as well as fine-tuned features from convolutional neural networks (CNN) should also be evaluated. We aim to extend this work beyond classification into e.g. localising appearance and subtle motion changes that are discriminative for action completion. An end-to-end CNN for detecting and localising incompletion is targetted.

Acknowledgements - The 1st author wishes to thank the University of Bristol for partial funding of her studies, and all the participants in the RGBD-Action-Completion-2016 data collection.

References

[1] G. Evangelidis, G. Singh, and R. Horaud. Skeletal quads: Human action recognition using joint quadruples. In *22nd International Conference on Pattern Recognition (ICPR)*, pages 4513 – 4518, 2014.

[2] J. Han, L. Shao, D. Xu, and J. Shotton. Enhanced computer vision with microsoft kinect sensor: A review. *IEEE Transactions on Cybernetics*, 43:1318–1334, 2013.

[3] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy. Sequence of the most informative joints (SMIJ): A new representation for human skeletal action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 8–13, 2012.

[4] O. Oreifej and Z. Liu. HON4D: Histogram of oriented 4D normals for activity recognition from depth sequences. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 716–723, 2013.

[5] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1297–1304, 2011.

[6] B. Soran, A. Farhadi, and L. Shapiro. Generating notifications for missing actions: Don't forget to turn the lights off! In *IEEE International Conference on Computer Vision (ICCV)*, pages 4669–4677, 2015.

[7] J. Sung, C. Ponce, B. Selman, and A. Saxena. Unstructured human activity detection from rgbd images. In *Robotics and Automation (ICRA), IEEE International Conference on*, pages 842–849, 2012.

[8] A. Vieira, E. Nascimento, G. Oliveira, Z. Liu, and M. Campos. STOP: Space-time occupancy patterns for 3D action recognition from depth map sequences. In *Pattern Recognition, Image Analysis, Computer Vision, and Applications (CIARP)*, pages 252–259, 2012.

[9] J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu. Robust 3D action recognition with Random Occupancy Patterns. In *Proceedings of the 12th European Conference on Computer Vision (ECCV)*, pages 872–885, 2012.

[10] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1290–1297, 2012.

[11] X. Wang, A. Farhadi, and A. Gupta. Actions ~ transformations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[12] Przemyslaw Woznowski, Xenofon Fafoutis, Terence Song, Sion Hannuna, Massimo Camplani, Lili Tao, Adeline Paiement, Evangelos Mellios, Mo Haghighi, Ni Zhu, et al. A multi-modal sensor infrastructure for healthcare in a residential environment. In *IEEE International Conference on Communications (ICC), Workshop on ICT-enabled services and technologies for eHealth and Ambient Assisted Living*, 2015.

[13] L. Xia and J. K. Aggarwal. Spatio-temporal depth cuboid similarity feature for activity recognition using depth camera. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2834–2841, 2013.

[14] L. Xia, C. Chen, and J. K. Aggarwal. View invariant human action recognition using histograms of 3D joints. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 20–27, 2012.

[15] X. Yang and Y. Tian. Effective 3D action recognition using eigenjoints. *Journal of Visual Communication and Image Representation.*, 25(1):2–11, 2014.

[16] X. Yang, C. Zhang, and Y. Tian. Recognizing actions using depth motion maps-based histograms of oriented gradients. In *Proceedings of the 20th ACM International Conference on Multimedia*, pages 1057–1060, 2012.

[17] A. Yao, J. Gall, and L. van Gool. Coupled action recognition and pose estimation from multiple views. *International Journal of Computer Vision (IJCV)*, 100:16–37, 2012.