SEQUENCE-TO-SEQUENCE MODELING FOR ACTION IDENTIFICATION AT HIGH TEMPORAL RESOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Automatic action identification from video and kinematic data is an important machine learning problem with applications ranging from robotics to smart health. Most existing works focus on identifying coarse actions such as running, climbing, or cutting a vegetable, which have relatively long durations. This is an important limitation for applications that require identification of subtle motions at high temporal resolution. For example, in stroke recovery, quantifying rehabilitation dose requires differentiating motions with sub-second durations. Our goal is to bridge this gap. To this end, we introduce a large-scale, multimodal dataset, *StrokeRehab*, as a new action-recognition benchmark that includes subtle short-duration actions labeled at a high temporal resolution. These short-duration actions are called motion primitives, and consist of reaches, transports, repositions, stabilizations, and idles. The dataset consists of high-quality Inertial Measurement Unit sensors and video data of 41 stroke-impaired patients performing activities of daily living like feeding, brushing teeth, etc. We show that current state-of-the-art models based on segmentation produce noisy predictions when applied to these data, which often leads to overcounting of actions. To address this, we propose a novel approach for high-resolution action identification, inspired by speech-recognition techniques, which is based on a sequence-to-sequence model that directly predicts the sequence of actions. This approach outperforms current state-of-the-art methods on the *StrokeRehab* dataset, as well as on the standard benchmark datasets: 50Salads, Breakfast, and Jigsaws.

1 INTRODUCTION

Automatic action identification from video and kinematic data is an important machine learning problem with applications ranging from robotics to smart health. Most previous works (e.g. TCN (Lea et al., 2017), MS-TCN (Farha & Gall, 2019), or ASRF (Ishikawa et al., 2021)) focus on identifying coarse actions such as running, climbing, sitting, putting bread on a plate, cutting a tomato, drinking from a glass, or inserting a needle, which usually have long durations. An important question is whether these approaches are effective in applications that require identification of short, single-goal motions at high temporal resolution, such as the rehabilitation of stroke patients. Identifying subtle, short-duration actions is key in data-driven stroke rehabilitation. Basic research in animals indicates that the repeated practice of functional motions early after stroke markedly boosts recovery (Jeffers et al., 2018; Bell et al., 2015; Murata et al., 2008). The same is believed to be true for humans undergoing rehabilitation for stroke-induced disability. However, there has been no systematic quantification of how many repetitions are needed early after stroke for optimal recovery (Krakauer et al., 2012). Data-driven quantification requires identifying and counting motions at high temporal (sub-second) resolution.

In order to advance methodology for high-resolution action identification, it is crucial to establish appropriate benchmark datasets. Existing benchmarks, such as 50Salads (Stein & McKenna, 2013), Breakfast (Kuehne et al., 2014), Jigsaws (Gao et al., 2014), Kinetics (Kay et al., 2017) or FineGym (Shao et al., 2020), contain very few short-duration actions (see Figure 3). To address this, we introduce a large-scale, multimodal dataset, *StrokeRehab*, as a new action-recognition benchmark that includes subtle short-duration actions labeled at a high temporal resolution. The dataset consists of high-quality wearable sensor and video data of 41 patients who are mildly or moderately impaired by stroke. These patients performed nine activities of daily living like drinking, eating,

applying deodorant, etc. in a rehabilitation gym. The subtle actions performed by the patients in each session were meticulously labeled by trained annotators overseen by an expert, who examined one-third of their labels.

Evaluation of current state-of-the-art approaches for action recognition (Farha & Gall, 2019; Ishikawa et al., 2021)) on the *StrokeRehab* dataset reveals a limitation of these methods for high-resolution action identification. These approaches are based on segmentation; they assign an action to each individual time step of the input data. It has been previously reported that this often yields noisy estimates (Ishikawa et al., 2021). Current methods address this issue by promoting smoothness of the segmentation output (Farha & Gall, 2019; Li et al., 2020) or by separately identifying the boundaries between actions (Ishikawa et al., 2021). We observe that these approaches are not as effective when applied to the short-duration subtle actions in the *StrokeRehab* dataset, where boundaries are not clearly defined even for human-expert annotators. As a result, segmentation-based approaches produce noisy estimates, which limits their accuracy. To address this limitation, we propose a novel approach to action identification inspired by speech-recognition models.

We propose a sequence-to-sequence (seq2seq) model that estimates the sequence of actions directly, without attempting to identify the boundaries between actions. The model learns to map variable-length input to variable-length output sequences using an encoder and a decoder. The encoder generates a hidden state vector from the input data, which is then decoded to generate an estimated sequence of actions. Such sequence-to-sequence approaches have been highly effective in speech recognition and natural language applications (Chan et al., 2015; Chiu et al., 2018; Prabhavalkar et al., 2017). The sequence-to-sequence model achieves state-of-the-art performance on the *StrokeRehab* dataset, and also outperforms existing approaches on the standard benchmark datasets like Breakfast, 50Salads, and Jigsaws for the task of sequence-prediction.

In summary, the contributions of our work are the following:

- We release a large-scale, multimodal dataset - *StrokeRehab* - which can be used for training and evaluating models to identify subtle actions at a high temporal resolution.
- We show that existing state-of-the-art methods based on segmentation generate noisy predictions when applied to the subtle short-duration actions in *StrokeRehab*, which leads to overcounting.
- Taking inspiration from speech recognition models, we propose a sequence-to-sequence method to predict a sequence of actions rather than individual, segmented actions. The proposed model outperforms the existing state-of-the-art methods on the *StrokeRehab* dataset and the benchmark datasets: 50Salads, Breakfast, and Jigsaws.

2 STROKE REHABILITATION DATASET

2.1 CLINICAL MOTIVATION

Stroke is the leading cause of disability in the United States. It affects nearly one million individuals per year, with the numbers of stroke cases increasing as our population ages (Go et al., 2014; Ovbiagele et al., 2013; Broderick, 2004). Stroke affects the arm in 77% of patients, causing long-lasting motor impairment (Lawrence et al., 2001; Kwakkel et al., 2015). By six months, most of these patients remain unable to independently perform activities of daily living, such as feeding, bathing, grooming, etc. This loss of independence reduces the quality of life of both the patients and their caretakers (Nichols-Larsen et al., 2005; Carod-Artal & Egido, 2009) and exacts a heavy societal toll, with caretaking and healthcare costs predicted to skyrocket to $240 billion by 2030 (Heidenreich et al., 2011). Due to the profound impact of the stroke on the arm and its downstream consequences, we focus on the arm in our study.

Following stroke, some spontaneous recovery occurs because of brain plasticity, but this plasticity alone does not fully restore function. In animal models of stroke, training high numbers of functional arm motions not only increases this plasticity (Kim et al., 2018; Bell et al., 2015), but also markedly boosts recovery (Murata et al., 2008; Jeffers et al., 2018). It is increasingly believed that if intensive rehabilitation training can be delivered early after stroke in humans, recovery could be similarly accelerated (Krakauer et al., 2012).

In rehabilitation training, patients use their impaired arm to practice activities of daily living (ADLs). ADLs are composed of five fundamental motions called functional primitives: reach, transport, reposition, stabilize, and idle (Schambra et al., 2019). For example, in a drinking activity, our arm would "idle" as it rests at our side, then "reach" for a glass and "transport" the glass to our mouth, then "transport" the glass back to the table, and finally "reposition" back to our side for an "idle." A visual example of primitives can be seen in Figure 1.

A major clinical question is how many repetitions of functional motions are needed for optimal recovery. In animal research, the number of motions that boost arm recovery has been quantified (Jeffers et al., 2018). For humans, this quantification has not been done. A handful of studies have observed that patients train about 10 times fewer repetitions than what recovering animals receive, suggesting pronounced under-training in human rehabilitation (Lang et al., 2009; Kimberley et al., 2010). However, the optimal number of training repetitions to boost recovery remains uncertain in humans. Currently, the best way to quantify rehabilitation is hand tallying. If performed in real time by the rehabilitation therapist, hand tallying distracts from treatment delivery. If the session is videotaped and annotated offline, tallying is laborious and slow: **it takes one hour of manual effort to label one minute of recorded training**. Hand tallying thus incurs time, effort, and personnel costs that render it unscalable.

Therefore, to facilitate the quantification of functional motions performed during stroke rehabilitation, we developed an approach that combines unobtrusive motion capture with automated identification. We collected the *StrokeRehab* dataset that consists of labeled sensor and video data from stroke patients. We then used the *StrokeRehab* dataset to train models to automatically identify and count functional primitives. We will make the complete dataset publicly available online upon publication (see here for sample data[1]).

2.2 Cohort selection

We collected sensor and video data from 41 stroke patients in an inpatient rehabilitation gym. Individuals were included if they were $\geq$ 18 years old, had premorbid right-handed dominance, and had unilateral arm weakness from ischemic or hemorrhagic stroke. Details about the inclusion and exclusion criteria and patient demography are provided in Appendix E.1.

2.3 Data Acquisition and Labelling

Upper body motion was recorded while the patients performed activities of daily living commonly used during stroke rehabilitation. The activities included: washing the face, applying deodorant, combing the hair, donning and doffing glasses, preparing and eating a slice of bread, pouring and drinking a cup of water, brushing teeth, and moving an object horizontal and vertical target arrays. See Appendix E.5 for detailed descriptions of the activities. The patients performed five repetitions of each activity.

Description of kinematic data: Upper body motion was recorded using nine Inertial Measurement Units (IMUs, Noraxon, USA) attached to the upper body, specifically the cervical vertebra C7, the thoracic vertebra T12, the pelvis, and both arms, forearms, and hands. These IMUs captured 76-dimensional kinematic features of 3D linear accelerations, 3D quaternions, and joint angles from the upper body (see Appendix E.2 and Appendix E.6 for details). As an additional feature, we included the paretic (stroke-affected) side of the patient (left or right) encoded in a one-hot vector, increasing the dimension of the feature vector to 77.

Description of video data: Video data were synchronously captured using two high definition cameras (1088 x 704, 60 frames per second; Ninox, Noraxon) placed orthogonally $<$ 2 m from the patient. We also extracted frame-wise feature vectors from the raw videos. The detailed procedure to extract these feature vectors is mentioned in Appendix E.3.

Data labeling: The subtle actions performed by the patients in each session were meticulously labeled by trained annotators overseen by an expert, who examined one-third of their labels. Interrater

[1] https://drive.google.com/drive/folders/1_a48XeRjFRdwaiaQXAV3tvAYb-4VmDbM?usp=sharing

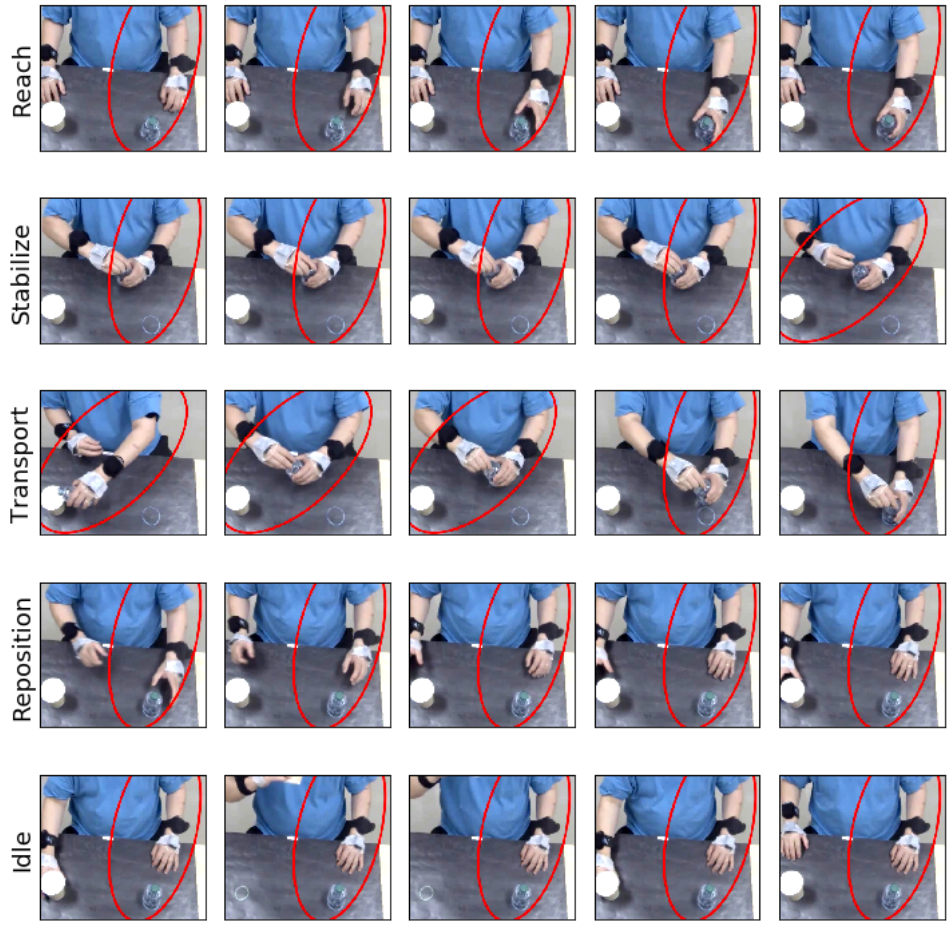

Figure 1: A stroke patient performing a functional activity (drinking) from the *StrokeRehab* activities battery. Using the functional motion taxonomy, the activity can be decomposed into its constituent functional primitives as follows: *reach*, upper extremity (UE) motion to bring it into contact with a target object (e.g. water bottle); *stabilize* minimal UE motion to keep a target object still (e.g. holding the bottle to allow the other UE to open the cap); *transport*, UE motion to move a target object (e.g. moving the bottle to pour some water); *reposition*, UE motion proximate to a target object (e.g. to move the to the initial neutral spot); *idle*, minimal UE motion to stand at the ready near a target object.

reliability between the coders and expert was high, with Cohen's kappa ≥ 0.96 between the coders and the expert. Details on the labeling procedure are provided in Appendix E.4

Train and test set: We assigned 33 patients to a training set and 8 patients to a test set. To this end, the 41 patients were separated into eight subgroups, balancing for impairment level and paretic side (left or right). One patient in each group was randomly removed and assigned to the test set. The remaining patients were assigned to the training set.

2.4 Data statistics and comparison to benchmark action recognition datasets

The dataset consists of 1,763 trials of rehabilitation activities performed by 41 stroke-impaired patients. Cumulatively, they performed 64284 functional primitives, making this dataset one of the largest in terms of the number of annotated actions. In comparison, the previous benchmark datasets like Breakfast (11,656 annotated actions), Jigsaws (1,701 annotated actions), and 50Salads (999 annotated actions) have a relatively low number of labeled actions. A non-trivial amount of manual effort (approximately 4,320 human hours) was required to label 64,284 fundamental primitives.

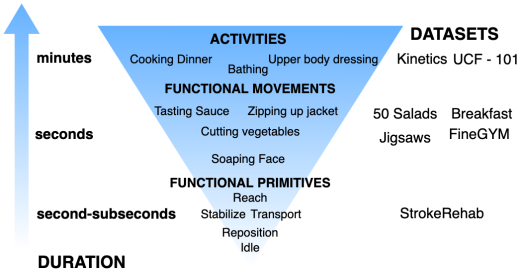

Figure 2: Popular action recognition datasets are aligned to a hierarchy of the labeled actions they contain. Our dataset *StrokeRehab* mainly contains short-duration basic functional primitives.

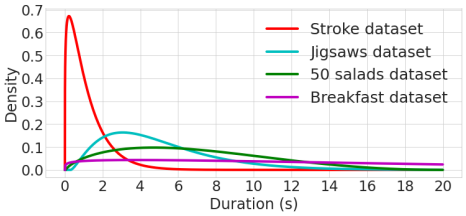

Figure 3: Distribution of action duration of actions for the various benchmark datasets. This illustrates the extreme fine-grained nature of actions (functional primitives) in the *StrokeRehab* dataset in comparison to existing ones.

In Figure 2, we show that the actions in *StrokeRehab* are at the bottom of the action hierarchy (Schambra et al., 2019). They are functional primitives with short durations that execute one goal. In existing datasets most actions are complex functional movements or activities, which have long durations and execute several goals. The difference in duration is evident in Figure 3. For the *StrokeRehab* dataset, most actions have a time duration of less than a second, whereas in the other datasets most actions take several seconds.

3 Relevant Work

Action classification: Action classification is popular for coarse actions identification, where instead of identifying a sequence of actions, only a single action needs to be identified from a sequence of video or sensor data. For video data, some previous methods (Wang et al., 2018; Lin et al., 2019; Zhou et al., 2018) have use 2D CNNs to extract frame-wise features from an input video, which are then combined to predict a coarse action taking place in the video. Since it is conceptually easy to consider a video as a 3D tensor with two spatial and one time dimension, C3D (Tran et al., 2015), I3D (Carreira & Zisserman, 2017) SlowFast (Feichtenhofer et al., 2019) and X3D (Feichtenhofer, 2020) use 3D CNNs to capture the abundant spatial-temporal information in video. There are also some prior works that conduct action classification using kinematic data. Early works use random forest or Hidden Markov Model (HMM) (Attal et al., 2015; Chung et al., 2019). Although these methods are relevant for coarse action identification, they are not relevant when it comes to identifying fine-grained actions. Unlike coarse actions, there are more than one fine-grained action in a single input data. Therefore, for identifying fine-grained actions, segmentation-based models are typically used.

Action segmentation: To provide fine-grained prediction, earlier approaches in action segmentation use sliding windows and filter out redundant hypotheses with non-maximum suppression (Wen & Keyes, 2019; Kaku et al., 2020). There are also some methods using a hidden Markov model (Kuehne et al., 2016) or RNN (Singh et al., 2016) to classify frame-wise actions. Inspired by the success of temporal convolutional network (TCN) in speech synthesis, Lea et al. (2017) introduces it for solving the task of action segmentation. MS-TCN (Farha & Gall, 2019) propose a multi-stage TCN architecture to help refine the prediction at each stage for reducing over-segmentation error. It has become the most widely used architecture for action segmentation. Though these works achieve high frame-wise accuracy, there are still significant over-segmentation errors. Considering this, there are some boundary-aware methods (Wang et al., 2020; Ishikawa et al., 2021) for temporally smoothing frame-wise predictions. These methods use the ground truth boundary information to train a binary classification network, and then use the boundary prediction to aggregate the frame-wise prediction either in a soft manner (boundary-aware pooling) or by setting a hard threshold. However, for the subtle actions with a high temporal resolution such as functional primitives, the duration of each action is very short. As a result, the boundaries between actions can be hard to detect or even hard to define.

Sequence-to-sequence model: Since the segmentation-based methods even with boundary refinement can cause over-segmentation errors for subtle actions with sub-second durations, we use

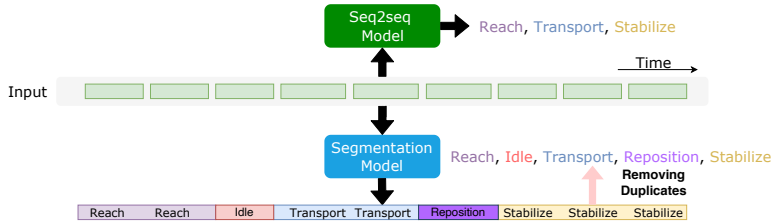

Figure 4: Comparison of sequence-to-sequence (seq2seq) and segmentation models. The segmentation model outputs frame-wise action predictions, which can then be converted to a sequence estimate by removing the duplicates. The seq2seq model produces a sequence estimate directly.

sequence-to-sequence (Seq2seq) models to directly predict the sequence of actions. Sequence-to-sequence model helps us to learn a mapping of a variable-length input sequence to a variable-length output sequence (Sutskever et al., 2014). There are several ways to build sequence-to-sequence models like RNN-Transducer (Graves, 2012), Connectionist temporal classification (CTC) based models (Graves et al., 2006) and encoder-decoder based models (Chan et al., 2015). We choose the encoder-decoder based models to perform the task of action sequence prediction as it is widely used for applications in machine translation (Bahdanau et al., 2014; Luong et al., 2014), speech recognition (Chan et al., 2015) and image captioning (Vinyals et al., 2015).

4 METHODS

We consider the problem of estimating a sequence of actions from a sequence of input data. Let $\mathbf{x} = (x_1, ..., x_T)$ be an input sequence with length T, which correspond to high-dimensional sensor or video features in our datasets of interest. We denote the corresponding sequence of actions by $\mathbf{y} = (y_1, ..., y_{T'})$, where y_i $(1 \leq i \leq T')$ is one of c different actions, so $y_i \in \{1, \ldots, c\}$. The length T' of this sequence is much shorter than the input sequence because each action takes place over several time steps. For example, a 100-frame video sequence from the *StrokeRehab* dataset may correspond to the 3-action sequence: Reach, Transport, Stabilize.

4.1 ACTION SEQUENCE PREDICTION BY ACTION SEGMENTATION MODEL

Most existing methods address the task of action sequence prediction by performing segmentation of the input data. Given $\mathbf{x}$, a segmentation model outputs frame-wise action labels $\mathbf{y_f}$ with the same length as the input. $\mathbf{y_f}$ can then be converted to $\mathbf{y}$ by removing label repetitions at consecutive steps (see Figure 4 for a concrete example). Segmentation-based models have been observed to systematically over-segment the input data (Farha & Gall, 2019; Wang et al., 2020; Ishikawa et al., 2021). Refs. (Wang et al., 2020; Ishikawa et al., 2021) address this by training a separate action-boundary detection network. The boundary predictions are then used to refine the frame-wise predictions. As pointed out by Shao et al. (2020), boundaries of high-level actions are more detectable because the adjacent motions are distinctive; for example, the motions associated with cutting tomatoes versus tossing a salad are very different. In contrast, boundaries of fine-grained actions are harder to identify because their transitions are more subtle; for example, the boundary between the end of a *reach* and the beginning of a *transport* in the *StrokeRehab* dataset is determined by when the finger pads have fully contacted a target object.

Figure 5 shows the accuracy of boundary-detection models for actions with different durations for several datasets. For all datasets, the accuracy of boundary detection is inversely proportional to duration. In addition, we observe that accurate boundary prediction can indeed be achieved for datasets with high-level actions (e.g. 50Salads and Breakfast), but is very challenging for the proposed dataset *StrokeRehab*. This suggests that segmentation-based approaches may be fundamentally limited for identification of subtle action sequences at high time resolution.

4.2 SEQUENCE-TO-SEQUENCE MODELING FOR ACTION IDENTIFICATION

Motivated by the limitations of segmentation-based approaches described in the previous section, we propose to directly predict the sequence of actions from the input data using sequence-to-sequence

(seq2seq) models inspired by speech recognition (see Figure 4). These models are designed to map input sequences to output sequences of different length, and are therefore well suited to the action-sequence estimation problem. The key idea is to *encode* the input data as a hidden vector of fixed dimension. The hidden vector is then *decoded* sequentially to produce the estimated sequence. The rest of this section explains the approach in detail.

Encoding: The encoder f_{enc}, which can be a convolutional or recurrent network, maps the input sequence $\mathbf{x}$ to a fixed-length hidden vector $\mathbf{h}(\theta_{\text{enc}}) = f_{\text{enc}}(\mathbf{x}; \theta_{\text{enc}})$ (θ_{enc} are the parameters of the encoder). The hidden vector must capture any long-term dependencies in the input sequence, which can be challenging in the case of high-level actions with long durations. For such cases, we have observed that using features from a pre-trained segmentation model can boost performance (see Section 5.2).

Decoding: The decoder is a recurrent neural network (RNN), which outputs the estimated sequence based on the hidden vector $\mathbf{h}$. For $i = 1, 2, \ldots$, the decoder estimates the conditional probability of the next action given the previous action and $\mathbf{h}$. To this end, the RNN f maintains a decoder state $\mathbf{s_i}$, which is updated based on the previous action and the hidden vector:

$$\mathbf{s_i}(\theta_{\text{dec}}) := f_{\text{dec}}(\mathbf{s_{i-1}}, \mathbf{h}, y_{i-1}; \theta_{\text{dec}}), \tag{1}$$

where θ_{dec} denotes the parameters of the decoder. For $i = 1$, the previous action is set to a *start-of-sequence* token. The conditional probability of y_i given the previous actions is then approximated using a multilayer perceptron (MLP) with a softmax output. Specifically,

$$\mathbf{p}_i(\theta_{\text{dec}}, \theta_{\text{enc}}, \theta_{\text{mlp}}) := \text{Softmax}(\text{MLP}(\mathbf{s_i}, \mathbf{h}; \theta_{\text{mlp}})) \tag{2}$$

is a $c+1$-dimensional vector that contains the estimates of the conditional probability that y_i equals each of the c possible actions or an *end-of-sequence* token. The ith predicted action $\hat{y}_i$ is obtained by maximizing this conditional probability. The procedure continues until $\hat{y}_i$ equals the *end-of-sequence* token. Optionally, we can incorporate an attention mechanism during decoding (Bahdanau et al., 2014; Chorowski et al., 2015).

Training: For the sake of simplicity, we consider a single training example $(\mathbf{x}, \mathbf{y})$, where the last entry of $\mathbf{y}$ contains the *end-of-sequence* token. The parameters of the encoder and decoder networks are learned by maximizing the objective function:

$$\max_{\theta_{\text{enc}}, \theta_{\text{dec}}, \theta_{\text{mlp}}} \sum_i \log(\mathbf{p}_i[y_i]), \tag{3}$$

where we omit the dependence of $\mathbf{p}_i$ on $\theta_{\text{enc}}, \theta_{\text{dec}}, \theta_{\text{mlp}}$ to ease notation. Here $\mathbf{p}_i[y_i]$ denotes the probability that the model assigns to the true observed action y_i when receiving $\mathbf{x}$ as an input.

During inference the ground-truth, $\mathbf{y}$ is not available, so the model can only use the previous predicted action $\hat{y}_{i-1}$ to compute $\mathbf{s_i}$ in equation 1. This suggests using the estimate also during training, a technique known as curriculum learning (Bengio et al., 2015). We apply this technique by replacing y_{i-1} with $\hat{y}_{i-1}$ in equation 1 with a probability ϵ, which is increased gradually during training.

Inference: During inference, we perform greedy-decoding to find the most likely sequence of actions given the input data. Specifically, at each time step i, we use the previous prediction, $\hat{y}_{i-1}$, in equation 1 to compute the decoder state, $\mathbf{s_i}$, which in turn is used to compute $\mathbf{p_i}$ using equation 2. Then, we choose $\arg\max_{1 \le j \le c+1} \mathbf{p_i}[j]$ as the predicted action for step i (note that there are $c+1$ possible actions because one is the *end-of-sequence* token). Beam-search decoding is often preferred over greedy-decoding in speech recognition and natural language applications (Chan et al., 2015), but here it did not provide a significant improvement.

5 EXPERIMENTS

In our experiments, we systematically compared the performance of our proposed sequence-to-sequence model to existing segmentation-based approaches on the *StrokeRehab* dataset described in Section 2, as well as the benchmark datasets 50Salads, Breakfast and Jigsaws (described in Appendix Appendix D.1).

5.1 PERFORMANCE METRICS

In order to evaluate sequence predictions we use two metrics based on the Levenshtein distance: edit score (ES) and action error rate (AER) (inspired by the word-error rate metric used in speech

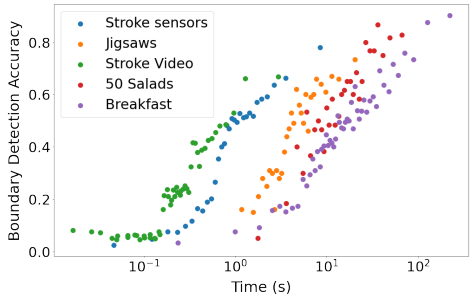

Figure 5: Boundary accuracy achieved by the segmentation models vs duration of the actions for several datasets. Boundary-detection accuracy is inversely proportional to action duration.

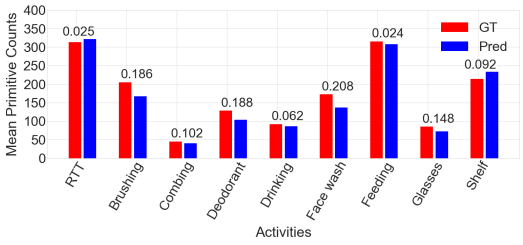

Figure 6: Comparison of ground-truth and predicted mean counts for the different activities in the *StrokeRehab* dataset. The relative error is very small for structured activities like moving objects on/off a shelf (Shelf), and larger for unstructured activities like brushing.

recognition). The Levenshtein distance, $\mathrm{L}(G, P)$, is the minimum number of insertions, deletions, and substitutions required to convert a predicted sequence P to a ground-truth sequence G. For example, if G = [Reach, Idle, Stabilize] and P = [Reach, Transport], then $\mathrm{L}(G, P) = 2$ ("Transport" is substituted for "Idle" and "Stabilize" is inserted). We have:

$$\mathrm{ES}(G, P) := \left(1 - \frac{\mathrm{L}(G, P)}{\max(\mathrm{len}(G), \mathrm{len}(P))}\right) \times 100, \qquad \mathrm{AER}(G, P) := \frac{\mathrm{L}(G, P)}{\mathrm{len}(G)} \tag{4}$$

where $\mathrm{len}(G)$ and $\mathrm{len}(P)$ are the lengths of the ground-truth and predicted sequence respectively.

The edit score is more lenient when the estimated sequence is longer. In contrast, AER penalizes longer and shorter predictions equally. For example, if G = [Reach, Idle, Stabilize], P_1 = [Reach,Idle], and P_2 = [Reach, Idle, Stabilize, Transport], then $\mathrm{ES}(G, P_1)$ = 0.67 and $\mathrm{ES}(G, P_2)$ = 0.75, but $\mathrm{AER}(G, P_1) = \mathrm{AER}(G, P_2)$ = 0.33.

5.2 Implementation Details of the Sequence-to-Sequence (Seq2seq) Model

During training we apply the seq2seq model on overlapping windows of short duration. During inference, we divide the input data into non-overlapping windows and concatenate the estimates removing duplicates at the boundaries. We implement two variants of the seq2seq method:

- *Raw2seq* uses raw sequences of sensor data or video features as input.
- *Seg2seq* uses frame-wise predictions from a segmentation-based model as inputs (specifically the MS-TCN baseline model in Section 5.3). The motivation is that the *raw2seq* has a limited field-of-view, but identifying high-level actions requires modeling long-term dependencies, which can be captured more easily by segmentation models.

Additional implementation details are provided in Appendix D.3

5.3 Segmentation Models

To ensure a fair comparison, we optimized our selected segmentation models using our metric of interest for segmentation prediction (AER). Detailed evaluation procedure can be seen in Section D.2.

Baseline: We use the MS-TCN (Farha & Gall, 2019) model with an improved loss function suggested in ASRF (Ishikawa et al., 2021) as a baseline model.

Baseline + boundary detection: This is the ASRF (Ishikawa et al., 2021) model, which is currently the state-of-the-art model for action segmentation. It refines the prediction of the baseline model using boundary predictions.

Baseline + smoothing window: We use a smoothing window to improve the baseline prediction.

	Model	Video Data		Sensor Data	
		Edit Score	Action Error Rate	Edit Score	Action Error Rate
Segmentation model	Baseline	62.2 (60.8 - 63.6)	0.392 (0.371 - 0.413)	68.9 (67.3 - 70.5)	0.330 (0.307 - 0.354)
	+ Boundary detection	58.7 (57.3 - 60.2)	0.436 (0.417 - 0.456)	67.9 (66.3 - 69.5)	0.349 (0.326 - 0.372)
	+ Smoothing window	62.7 (61.3 - 64.1)	0.390 (0.370, 0.410)	68.8 (67.3 - 70.3)	0.317 (0.297 - 0.338)
Seq2seq	Seg2seq	**67.6 (66.4 - 68.8)**	**0.322 (0.307 - 0.339)**	63.0 (61.3 - 64.7)	0.337 (0.311 - 0.363)
	Raw2seq	66.6 (65.4 - 67.9)	0.329 (0.312 - 0.345)	**69.0 (67.3 - 70.6)**	**0.308 (0.287 - 0.329)**

Table 1: Results on *StrokeRehab*: Seq2seq outperforms segmentation-based approaches. We report mean (95% confidence interval) which is computed via bootstrapping the test set. The bootstrapping details are mentioned in Appendix D.2.

	Model	50 salads		Breakfast		Jigsaws	
		Edit Score	Action Error Rate	Edit Score	Action Error Rate	Edit Score	Action Error Rate
Segmentation model	Baseline	70.8	0.43	61.7	0.97	61.44	0.82
	+ Boundary detection	75.2	0.33	70.9	0.45	74.63	0.31
	+ Smoothing window	76.4	0.32	69.1	0.51	76.54	0.31
Seq2seq	Seg2seq	**76.9**	**0.30**	**73.7**	**0.37**	**83.87**	**0.17**
	Raw2seq	69.4	0.54	64.1	0.55	70.13	0.35

Table 2: Results on action-recognition benchmarks: Seg2seq, the seq2seq model which uses the output of a pretrained segmentation-based model, outperforms segmentation-based approaches.

5.4 Results

Table 1 and Table 2 show that sequence-to-sequence models outperform segmentation-based models on all the datasets. The raw2seq version achieves better performance on the sensor data of the *StrokeRehab* data, where the actions are very localized and do not require modeling long time dependencies, whereas the seg2seq version is superior on the remaining datasets. The baselines achieve edit score values that are close to those of seq2seq on the sensor data of *StrokeRehab* (but not on the video data), and on 50Salads, but the AER of seq2seq is better (as explained in Section 5.1 the edit score is more lenient with the false positives that tend to be produced by segmentation-based models). Interestingly, the refinement strategies of smoothing and boundary detection do not improve the baseline on the *StrokeRehab*, highlighting that it contains actions that qualitatively different from those of the other datasets.

5.5 Application: Data-driven Stroke Rehabilitation

In stroke rehabilitation, action identification can be used for quantifying dose by counting functional primitives. Figure 6 show that the raw2seq version of the seq2seq model produces accurate counts for all activities in the *StrokeRehab* dataset. Performance is particularly good for structured activities such as moving objects on/off a shelf, in comparison to less structured activities such as brushing, which tend to be more heterogeneous across patients (Detailed count-based results can be seen in Appendix B and Appendix C). Figure 7 and Figure 8 provide a detailed analysis of the predictions produced by the best models based on video and sensor data respectively. The sensor-based model has difficulties differentiating between *idle* and *transport*, whereas the video-based model confuses *idle* and *reach*.

6 Discussion and Conclusion

In this work, we introduce a large-scale, multimodal dataset, *StrokeRehab*, as a new benchmark for identification of action sequences that includes subtle short-duration actions labeled at a high temporal resolution. In addition, we introduce a novel sequence-to-sequence approach, which outperforms existing methods on *StrokeRehab*, as well as on existing benchmark datasets. A known limitation of sequence-to-sequence approaches is that they have difficulties capturing long-term dependencies. Here, we address this by using the output of a segmentation-based network as an input for the sequence-to-sequence model. An interesting direction for future research is to design sequence-to-sequence models capable of directly learning these dependencies. Our results also show that models based on video and wearable-sensor data have different strengths and weaknesses (see Section 5.5), which suggests that multimodal approaches may have significant potential.

REFERENCES

Ferhat Attal, Samer Mohammed, Mariam Dedabrishvili, Faicel Chamroukhi, Latifa Oukhellou, and Yacine Amirat. Physical human activity recognition using wearable sensors. *Sensors*, 15(12): 31314–31338, 2015.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Jared A Bell, Malerie L Wolke, Ryan C Ortez, Theresa A Jones, and Abigail L Kerr. Training intensity affects motor rehabilitation efficacy following unilateral ischemic insult of the sensorimotor cortex in c57bl/6 mice. *Neurorehabilitation and neural repair*, 29(6):590–598, 2015.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *arXiv preprint arXiv:1506.03099*, 2015.

Joseph P Broderick. William m. feinberg lecture: stroke therapy in the year 2025: burden, breakthroughs, and barriers to progress. *Stroke*, 35(1):205–211, 2004.

Francisco Javier Carod-Artal and José Antonio Egido. Quality of life after stroke: the importance of a good recovery. *Cerebrovascular diseases*, 27(Suppl. 1):204–214, 2009.

Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6299–6308, 2017.

William Chan, Navdeep Jaitly, Quoc V Le, and Oriol Vinyals. Listen, attend and spell. *arXiv preprint arXiv:1508.01211*, 2015.

Chung-Cheng Chiu, Tara N Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J Weiss, Kanishka Rao, Ekaterina Gonina, et al. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4774–4778. IEEE, 2018.

Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. *arXiv preprint arXiv:1506.07503*, 2015.

Seungeun Chung, Jiyoun Lim, Kyoung Ju Noh, Gague Kim, and Hyuntae Jeong. Sensor data acquisition and multimodal sensor fusion for human activity recognition using deep learning. *Sensors*, 19(7):1716, 2019.

Yazan Abu Farha and Jurgen Gall. Ms-tcn: Multi-stage temporal convolutional network for action segmentation. In *In proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3575–3584, 2019.

Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 203–213, 2020.

Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Inproceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6202–6211, 2019.

Axel R Fugl-Meyer, L Jääskö, Ingegerd Leyman, Sigyn Olsson, and Solveig Steglind. The post-stroke hemiplegic patient. 1. a method for evaluation of physical performance. *Scandinavian journal of rehabilitation medicine*, 7(1):13–31, 1975.

Yixin Gao, S Swaroop Vedula, Carol E Reiley, Narges Ahmidi, Balakrishnan Varadarajan, Henry C Lin, Lingling Tao, Luca Zappella, Benjamın Béjar, David D Yuh, et al. Jhu-isi gesture and skill assessment working set (jigsaws): A surgical activity dataset for human motion modeling. In *MICCAI workshop: M2cai*, volume 3, pp. 3, 2014.

Alan S Go, Dariush Mozaffarian, Véronique L Roger, Emelia J Benjamin, Jarett D Berry, Michael J Blaha, Shifan Dai, Earl S Ford, Caroline S Fox, Sheila Franco, et al. Executive summary: heart disease and stroke statistics—2014 update: a report from the american heart association. *Circulation*, 129(3):399–410, 2014.

Alex Graves. Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*, 2012.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pp. 369–376, 2006.

Paul A Heidenreich, Justin G Trogdon, Olga A Khavjou, Javed Butler, Kathleen Dracup, Michael D Ezekowitz, Eric Andrew Finkelstein, Yuling Hong, S Claiborne Johnston, Amit Khera, et al. Forecasting the future of cardiovascular disease in the united states: a policy statement from the american heart association. *Circulation*, 123(8):933–944, 2011.

Yuchi Ishikawa, Seito Kasai, Yoshimitsu Aoki, and Hirokatsu Kataoka. Alleviating oversegmentation errors by detecting action boundaries. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2322–2331, 2021.

Matthew Strider Jeffers, Sudhir Karthikeyan, Mariana Gomez-Smith, Sarah Gasinzigwa, Jannis Achenbach, Astrid Feiten, and Dale Corbett. Does stroke rehabilitation really matter? part b: an algorithm for prescribing an effective intensity of rehabilitation. *Neurorehabilitation and neural repair*, 32(1):73–83, 2018.

Aakash Kaku, Avinash Parnandi, Anita Venkatesan, Natasha Pandit, Heidi Schambra, and Carlos Fernandez-Granda. Towards data-driven stroke rehabilitation via wearable sensors and deep learning. In *Machine Learning for Healthcare Conference*, pp. 143–171. PMLR, 2020.

Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

Soo Young Kim, J Edward Hsu, Lincoln C Husbands, Jeffrey A Kleim, and Theresa A Jones. Coordinated plasticity of synapses and astrocytes underlies practice-driven functional vicariation in peri-infarct motor cortex. *Journal of Neuroscience*, 38(1):93–107, 2018.

Teresa Jacobson Kimberley, Sharyl Samargia, Lisa G Moore, Josefin K Shakya, and Catherine E Lang. Comparison of amounts and types of practice during rehabilitation for traumatic brain injury and stroke. 2010.

John W Krakauer, S Thomas Carmichael, Dale Corbett, and George F Wittenberg. Getting neurorehabilitation right: what can be learned from animal models? *Neurorehabilitation and neural repair*, 26(8):923–931, 2012.

Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 780–787, 2014.

Hilde Kuehne, Juergen Gall, and Thomas Serre. An end-to-end generative framework for video segmentation and recognition. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1–8. IEEE, 2016.

Gert Kwakkel, Janne M Veerbeek, Erwin EH van Wegen, and Steven L Wolf. Constraint-induced movement therapy after stroke. *The Lancet Neurology*, 14(2):224–234, 2015.

Catherine E Lang, Jillian R MacDonald, Darcy S Reisman, Lara Boyd, Teresa Jacobson Kimberley, Sheila M Schindler-Ivens, T George Hornby, Sandy A Ross, and Patricia L Scheets. Observation of amounts of movement practice provided during stroke rehabilitation. *Archives of physical medicine and rehabilitation*, 90(10):1692–1698, 2009.

Enas S Lawrence, Catherine Coshall, Ruth Dundas, Judy Stewart, Anthony G Rudd, Robin Howard, and Charles DA Wolfe. Estimates of the prevalence of acute stroke impairments and disability in a multiethnic population. *Stroke*, 32(6):1279–1284, 2001.

Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 156–165, 2017.

Shi-Jie Li, Yazan AbuFarha, Yun Liu, Ming-Ming Cheng, and Juergen Gall. Ms-tcn++: Multi-stage temporal convolutional network for action segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *In proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7083–7093, 2019.

Minh-Thang Luong, Ilya Sutskever, Quoc V Le, Oriol Vinyals, and Wojciech Zaremba. Addressing the rare word problem in neural machine translation. *arXiv preprint arXiv:1410.8206*, 2014.

Yumi Murata, Noriyuki Higo, Takao Oishi, Akiko Yamashita, Keiji Matsuda, Motoharu Hayashi, and Shigeru Yamane. Effects of motor training on the recovery of manual dexterity after primary motor cortex lesion in macaque monkeys. *Journal of neurophysiology*, 99(2):773–786, 2008.

Deborah S Nichols-Larsen, PC Clark, Angelique Zeringue, Arlene Greenspan, and Sarah Blanton. Factors influencing stroke survivors' quality of life during subacute recovery. *Stroke*, 36(7):1480–1484, 2005.

Bruce Ovbiagele, Larry B Goldstein, Randall T Higashida, Virginia J Howard, S Claiborne Johnston, Olga A Khavjou, Daniel T Lackland, Judith H Lichtman, Stephanie Mohl, Ralph L Sacco, et al. Forecasting the future of stroke in the united states: a policy statement from the american heart association and american stroke association. *Stroke*, 44(8):2361–2375, 2013.

Rohit Prabhavalkar, Kanishka Rao, Tara N Sainath, Bo Li, Leif Johnson, and Navdeep Jaitly. A comparison of sequence-to-sequence models for speech recognition. In *Interspeech*, pp. 939–943, 2017.

Heidi M Schambra, Avinash R Parnandi, Natasha G Pandit, Jasim Uddin, Audre Wirtanen, and Dawn M Nilsen. A taxonomy of functional upper extremity motion. *Frontiers in neurology*, 10: 857, 2019.

Dian Shao, Yue Zhao, Bo Dai, and Dahua Lin. Finegym: A hierarchical video dataset for fine-grained action understanding. In *In proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2616–2625, 2020.

Bharat Singh, Tim K Marks, Michael Jones, Oncel Tuzel, and Ming Shao. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1961–1970, 2016.

Sebastian Stein and Stephen J McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pp. 729–738, 2013.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pp. 3104–3112, 2014.

Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 4489–4497, 2015.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3156–3164, 2015.

Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *IEEE transactions on pattern analysis and machine intelligence*, 41(11):2740–2755, 2018.

Zhenzhi Wang, Ziteng Gao, Limin Wang, Zhifeng Li, and Gangshan Wu. Boundary-aware cascade networks for temporal action segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 34–51. Springer, 2020.

Tailai Wen and Roy Keyes. Time series anomaly detection using convolutional neural networks and transfer learning. *arXiv preprint arXiv:1905.13628*, 2019.

Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 803–818, 2018.

A CONFUSION MATRIX FOR DIFFERENT MODALITIES OF *StrokeRehab* DATASET

We study the differences between the models trained on different modalities of the *StrokeRehab*. To this end, we look at the confusion matrix to understand the errors made by the two models: *Raw2Seq* on the sensor data and *Seg2seq* on the video data. The confusion matrix is built using the total number of substitutions edits and correct predictions from both models. Specifically, each diagonal cell of the confusion matrix represents the fraction of ground-truth primitives that we got correct and each non-diagonal cell represents the fraction of ground-truth primitive that were substituted. We found that the nature of errors of two models were non-overlapping. For example, from Figure 7 and Figure 8, we can see that the model trained on sensor data found it comparatively more difficult to differentiate between idles and transports whereas the model trained on video data found idles to be similar to reaches. Since the nature of errors is disparate, this makes a good case for training multi-modal models that can leverage both modalities to perform the task of sequence estimation.

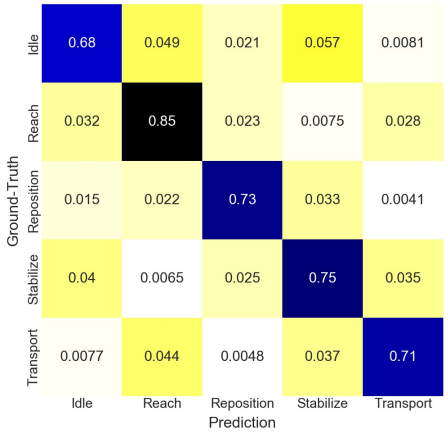

Figure 7: Confusion Matrix for the best performing model on the *StrokeRehab* video Dataset. The model makes the most mistake for Idle action.

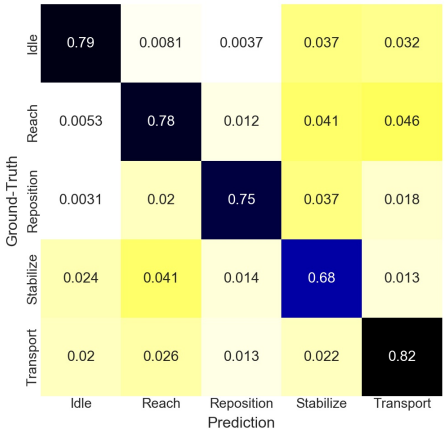

Figure 8: Confusion Matrix for the best performing model on the *StrokeRehab* sensor Dataset. The model makes the most mistake for Stabilize action.

B FALSE DISCOVERY RATE AND TRUE POSITIVE RATE

In addition to edit score and action error rate, we also evaluated the model performance using the true positive rate (TPR) and the false discovery rate (FDR). To compute these quantities we compared the estimated sequence to the ground truth actions to determine the number of actions that are correctly identified, incorrectly identified (e.g. the model estimates a transport, but the ground truth is a reach), missed (an action present in the ground truth is not present in the estimate), or spurious (an action present in the estimate is not present in the ground truth). The TPR is the ratio between the correctly-identified actions and the total ground-truth actions. The FDR is the ratio between the wrong predictions (incorrectly identified and spurious) and the total estimated actions. It should be noted that there is a trade-off between TPR and FDR. Therefore, if we want to ascertain superiority of a model, both TPR and FDR should be combined using F1-score which can then be used to compare different models. F1-score can be calculated as follows: F1-score $= \frac{2(1-FDR)TPR}{1-FDR+TPR}$

	Model	Video Data		Sensor Data	
		FDR	TPR	FDR	TPR
Segmentation model	Baseline	0.148 (0.135 - 0.161)	0.643 (0.622- 0.664)	0.201 (0.186 - 0.216)	0.790 (0.770 - 0.812)
	+ Boundary detection	0.126 (0.113 - 0.139)	0.585 (0.566 - 0.605)	0.219 (0.205 - 0.233)	0.784 (0.763 - 0.803)
	+ Smoothing window	0.180 (0.166 - 0.194)	0.666 (0.646 - 0.685)	0.162 (0.149 - 0.173)	0.758 (0.737 - 0.779)
Seq2seq	Seg2seq	0.211 (0.200 - 0.221)	0.743 (0.732 - 0.753)	0.145 (0.134 - 0.157)	0.707 (0.678 - 0.734)
	Raw2seq	0.216 (0.207 - 0.226)	0.734 (0.722 - 0.744)	0.166 (0.153 - 0.179)	0.767 (0.747 - 0.786)

Table 3: Comparison of seq2seq model and action segmentation model on *StrokeRehab*: in terms of FDR and TPR

	Model	50 salads		Breakfast		Jigsaws	
		FDR	TPR	FDR	TPR	FDR	TPR
Segmentation model	Baseline	0.26	0.86	0.34	0.82	0.357	0.898
	+ Boundary detection	0.17	0.80	0.19	0.77	0.191	0.846
	+ Smoothing window	0.18	0.83	0.23	0.79	0.174	0.878
Seq2seq	Seg2seq	0.17	0.82	0.17	0.79	0.092	0.874
	Raw2seq	0.31	0.76	0.28	0.70	0.218	0.780

Table 4: Comparison of seq2seq model and action segmentation model on 50Salads, breakfast, Jigsaws: in terms of FDR and TPR

C COUNT PREDICTION: SUBJECT-WISE

Subject	Ground-Truth					Prediction				
	Idle	Reach	Reposition	Transport	Stabilize	Idle	Reach	Reposition	Transport	Stabilize
s4	397	194	290	415	301	293	213	204	370	281
s17	282	353	257	489	567	252	348	225	582	724
s26	237	293	211	245	319	240	253	211	183	326
s37	188	337	191	266	464	202	328	174	113	408
s39	338	273	250	228	294	324	252	219	389	366
s42	290	328	245	330	470	287	297	228	292	417
s44	298	365	264	294	429	285	275	229	327	406
s47	215	313	203	186	427	181	287	156	74	332
Mean	280.62	307	238.87	306.62	408.87	258	281.62	205.75	291.25	407.5
Std-Dev	63.26	51.21	31.8	94.8	90.43	45.52	40.73	25.24	155.02	127.64

Table 5: The ground-truth and the predicted (by the best performing model) number of the different functional primitives across the patients in the test set on the *StrokeRehab* sensor data

Subject	Ground-Truth					Prediction				
	Idle	Reach	Reposition	Transport	Stabilize	Idle	Reach	Reposition	Transport	Stabilize
s4	151	282	229	363	307	153	402	123	416	160
s17	336	541	224	466	261	263	634	231	493	266
s26	293	315	202	235	221	247	320	195	208	199
s37	251	356	95	219	90	201	369	84	239	78
s39	142	165	128	182	205	119	199	125	212	162
s42	219	351	136	291	157	188	309	120	258	156
s44	223	282	143	193	160	151	227	147	155	156
s47	198	304	98	174	79	162	242	99	181	83
Mean	226.62	324.5	156.87	265.37	185	185.5	337.75	140.5	270.25	157.5
Std-Dev	61.98	98.82	50.62	96.21	74.18	46.44	129.76	46.21	112.08	56.34

Table 6: The ground-truth and the predicted (by the best performing model) number of the different functional primitives across the patients in the test set on the *StrokeRehab* video data

Based on Tables 5 and 6, we observe that the predicted counts of the functional primitives are close to their corresponding ground truth counts, which means our models perform well on quantifying stroke rehab and therefore can help in large scale studies that seek to find the optimum rehabilitation. Note that the number of ground-truth actions for sensor dataset and video dataset are different. We filter out some data in the preprocess procedure. Therefore, there exists some sequences which only have single modality.

D IMPLEMENTATION DETAILS

D.1 BENCHMARK DATASETS

50 salads dataset 50 salads dataset (Stein & McKenna, 2013) contains 50 videos with 17 action classes. In this dataset, 25 people in total are preparing two kinds of mixed salads. Each video contains 9000 to 19000 RGB frames. On average, each video contains 20 action instances and is 6.4 minutes long. For evaluation, we use the same protocol as the ASRF model by five-fold cross-validation and report the average.

Breakfast dataset Breakfast dataset (Kuehne et al., 2014) contains 1712 videos, which are recorded in 18 different kitchens displaying activities of breakfast preparation. There are 48 different actions. On average, each video contains 6 action instance. For evaluation, we use the same protocol as the ASRF model by four-fold cross-validation and report the average.

Jigsaws dataset Jigsaws dataset (Gao et al., 2014) contains 103 surgical activities of 3 types - Knot tying, Suturing and Needle Passing performed by 8 subjects using a robotic surgical system. The subjects have varying degrees of robotic surgical experience. The kinematic data from the robotic arm and the videos of the activities are available. The activities contain 14 different actions overall. On average, each video contains 17 action instances. We split the subjects into 4 folds,where each fold contains two subjects of different robotic experience levels. For evaluation, we use the same protocol as the ASRF model by four-fold cross-validation and report the average.

D.2 EVALUATION

For the action segmentation model, we follow the setting in ASRF (Ishikawa et al., 2021) for Breakfast dataset (Kuehne et al., 2014), 50Salads dataset (Stein & McKenna, 2013) and Jigsaws dataset (Gao et al., 2014). We use a separate small validation set divided from the training set of each fold to select models based on Action Error Rate (AER), and test the performance on the val set of that fold. We then following the same evaluation as the previous methods (Farha & Gall, 2019; Ishikawa et al., 2021) by simply averaging the result for all the folds. For the *StrokeRehab*, there is a common leave out test set for all the 4 folds. For each fold, we select the model according to the best Action Error Rate (AER) on the validation set and test the performance on the leave out test set. We ensemble the prediction of the models trained on 4 folds on the common test set. For the action segmentation model prediction, we take the average of the probability of the 4 folds. For the seq2seq model, the decoding on the next action depends on the prediction of the previous step. At each step, we feed the average predictions of the four models from the previous step to individual model for more accurate decoding. We then take the averaged result as the final result.

Confidence interval computation: We report the 95% confidence interval for the *StrokeRehab* dataset by bootstrapping the test set. Specifically, we create 1000 bootstrap replicates of the test set. We calculate the upper and lower confidence limit using $\mu \pm 1.96\sigma/\sqrt{N}$ where μ and σ are mean and standard deviation of the performance measure for 1000 bootstrap replicates, respectively. N is number of bootstrap replicates.

D.3 IMPLEMENTATION OF MODELS

D.3.1 ACTION SEGMENT REFINEMENT FRAMEWORK (ASRF)

The ASRF model has two modules: one for frame-wise action segmentation and the other for boundary detection. It uses a long-term feature extractor to capture the long range feature dependency, based on which they have a separate action segmentation module and a boundary detection module to get framewise action predictions and action boundaries. The long-term feature extractor is a MS-TCN model that has four stages of convolutions with each stage having ten layers of dilated residual convolutions outputting 64 channels. The model is trained on the raw sensor data as the features. We input the entire session/activity to model without windowing it. During inference, we have two ways to generate the predictions. In the first way, we just take the output of the segmentation module as the predictions (Baseline model in Table 1 and Table 2). In the second way, we refine the outputs of

the segmentation module using the boundaries detected by the boundary detection module (Baseline + Boundary Detection model in Table 1 and Table 2).

The loss function used to train this model is combination of two loss functions: weight-cross entropy for frame-wise action classification and boundary detection. We use a parameter lambda to determine the weight of boundary detection loss. For sensor dataset, the lambda is set to 0.1. For the stroke video dataset, the lambda is set to be 1.

For the 50Salads and breakfast dataset, we follow the similar setting as the original paper. The only difference is that we are selecting the model on the validation performance of AER rather than the frame-wise accuracy. We do this with an aim to achieve a fair comparison with the seq2seq model on the action order prediction performance.

For the Jigsaws dataset, we used the 38-dimension kinematic sensor data as input. In the ASRF, MSTCN model has 4 stages of convolutions with each stage having 15 layers of dilated residual convolutions outputting 128 channels. The weights for the boundary detection loss was 0.1.

D.3.2 SEQUENCE-TO-SEQUENCE (SEQ2SEQ)

All the following Seq2seq models are trained with the following hyper-parameters: learning rate = 5e-4, dropout = 0.1, weight decay = 0.0001, num of epochs = 150. When dividing the whole sequence to time windows, if the last sequence is shorter than the required time window size, zero padding is applied to ensure equal length.

Seq2seq for *StrokeRehab* video dataset:

For *StrokeRehab* video dataset, the encoder module is based on MS-TCN architecture with 256 dimensional hidden representation. The decoder module is an LSTM with attention mechanism using 512 dimensional hidden representation.

- *Raw2seq*. During training, we divide each video sequence of 432 dimensional raw feature vectors into 240-frame windows with overlap of 100 frames. We then train the model using the windows. During inference, we divide each testing video sequence to non-overlapping 240-frame windows and input that to the model. We then concatenate the result from the non-overlapping window and remove the duplicates.
- *Seg2seq*. We first train a ASRF segmentation model to obtain raw frame-wise prediction probability without refinement, with the 4 stage MS-TCN architecture (4 stacked single stage TCNs). We take the frame-wise softmax probabilities as the input to the Seq2seq model. During training, we divide softmax probabilities of each video sequence to 240-frame windows with overlap of 100 frames. We then train the model using the windowed data. During inference, we divide each testing video sequence to non-overlapping 240-frame windows and input that to the model. We then concatenate the result from the non-overlapping window and remove the duplicates.

Seq2seq for *StrokeRehab* sensor dataset:

- *Raw2seq*. We use 77 dimensional raw sensor readings and feed it to the seq2seq model. Seq2seq model has an encoder module and decoder module. The encoder module is a three layered bi-LSTM model with 3072 dimensional hidden representation. The decoder module is also an LSTM with 6144 dimensional hidden representation. To facilitate the training of the model, we window the data and then input it to model. During training, we input overlapping six-seconds windows to the model with labels being determined by the middle four seconds. During inference, we input non-overlapping windows to the model. We then concatenate the result from the non-overlapping window and remove the duplicates. We use a primitive wise as well as frame-wise cross-entropy loss to train the seq2seq model. We also use a dot product based attention module.
- *Seg2seq*. Here, we take the frame-wise softmax probabilities of the segmentation module and feed it to the seq2seq model. Seq2seq model has an encoder module and decoder module. The encoder module is a three layered bi-LSTM model with 256 dimensional hidden representation. The decoder module is also an LSTM with 512 dimensional hidden representation. To facilitate the training of the model, we window the data and then input it to model. During training, we input overlapping six-seconds windows to the model with labels being determined by the middle four seconds. During inference, we input non-overlapping windows to the model. We then concatenate

the result from the non-overlapping window and remove the duplicates. While decoding, we also use a dot product based attention module. We use a primitive wise as well as frame-wise cross-entropy loss to train the seq2seq model.

Seq2seq for 50Salads dataset:

- *Raw2seq*. During training, we divide each video sequence of 1600 dimensional raw feature vectors to 500-frame windows with overlap of 100 frames. We then train the model using the windows. During inference, we divide each testing video sequence to non-overlapping 1600-frame windows and input that to the model. We then concatenate the result from the non-overlapping window and remove the duplicates.
- *Seg2seq*. We first train an ASRF segmentation model to obtain raw frame-wise prediction probability without refinement, with the 4 stage MS-TCN architecture (4 stacked single stage TCNs). We take the frame-wise softmax probabilities as the input to the Seq2seq model. Seq2seq model has an encoder module and a decoder module. During training, we divide softmax probabilities of each video sequence to 450-frame windows with overlap of 100 frames. We then train the model using the windowed data. During inference, we divide each testing video sequence to non-overlapping 450-frame windows and input that to the model. We then concatenate the result from the non-overlapping window and remove the duplicates.

Seq2seq for Breakfast dataset:

- *Raw2seq*. During training, we divide each video sequence of 1600 dimensional raw feature vectors to 500-frame windows with overlap of 100 frames. We then train the model using the windows. During inference, we divide each testing video sequence to non-overlapping 1600-frame windows and input that to the model. We then concatenate the result from the non-overlapping window and remove the duplicates.
- *Seg2seq*. We first train a ASRF segmentation model to obtain raw frame-wise prediction probability without refinement, with the 4 stage MS-TCN architecture (4 stacked single stage TCNs). We take treat the frame-wise softmax probabilities as the input to the Seq2seq model. Seq2seq model has an encoder module and a decoder module. During training, we divide softmax probabilities of each video sequence to 800-frame windows with overlap of 200 frames. We then train the model using the windowed data. During inference, we divide each testing video sequence to non-overlapping 800-frame windows and input that to the model. We then concatenate the result from the non-overlapping window and remove the duplicates.

Seq2seq for Jigsaws dataset:

- *Raw2seq*. Seq2seq for raw features of Jigsaws has an encoder, a decoder module and attention mechanism. The encoder is a MSTCN model with 4 stages of convolutions with each stage having 10 layers of dilated residual convolutions outputting 128 channels. The decoder is a GRU-RNN with 256 dimensional hidden representation. We also used a multi-headed attention mechanism (2 heads) with a multi-layer perceptron producing 128 dimensional representation. During training, we divide each Kinematic input sequence of 38 dimensional raw feature vectors to 400-frame windows with an overlap of 350 frames. We then train the model using these windows. During inference, we divide each testing video sequence to non-overlapping 400-frame windows and input that to the model. We then concatenate the result from the non-overlapping window and remove the duplicates.
- *Seg2seq*. Seq2seq with action segmentation model has an encoder, a decoder module and attention mechanism. The encoder is a MS-TCN model with 4 stages of convolutions with each stage having 5 layers of dilated residual convolutions outputting 64 channels. The decoder is a LSTM with 256 dimensional hidden representation. We also used a single-headed attention mechanism with a multi-layer perceptron producing 64 dimensional representation. We first train a ASRF segmentation model to obtain raw frame-wise prediction probability without refinement, with the 4 stage MS-TCN architecture (4 stacked single stage TCNs). We treat the frame-wise softmax probabilities as the input to the Seq2seq model. Seq2seq model has an encoder module and a decoder module. During training, we input the entire input sequence to model without windowing it. The seq2seq model is trained with a primitive level cross-entropy loss. During inference,we input the entire input sequence to model without windowing it and remove the duplicates.

E *StrokeRehab* DATASET DESCRIPTION

E.1 COHORT SELECTION

Individuals were excluded if they had traumatic brain injury; any musculoskeletal or non-stroke neurological condition that interferes with motor function; contracture at the shoulder, elbow, or wrist; moderate arm dysmetria or truncal ataxia; visuospatial neglect; apraxia; global inattention; or legal blindness. A trained assessor quantified arm impairment with the upper extremity Fugl-Meyer Assessment, where a maximum score of 66 signifies no impairment (Fugl-Meyer et al., 1975). Patients were moderately to mildly impaired (scores 26-65). Table 7 describes the demographic and clinical characteristics of the patients.

	Training set	Test set
n	33	8
Age (in years)	56.3 (21.3-84.3)	60.9 (42.6-84.3)
Gender	18 F : 15 M	4 F : 4 M
Time since stroke (in years)	6.5 (0.3-38.4)	3.1 (0.4-5.7)
Paretic side (Left : Right)	18 L : 15 R	4 L : 4 R
Stroke type (Ischemic : Hemorrhagic)	30 I : 3 H	8 I : 0 H
Fugl-Meyer Assessment score	48.1 (26-65)	49.4 (27-63)

Table 7: Demographic and clinical characteristics of the patients in the cohort. Mean and ranges in parentheses are shown. The cohort is divided into a training set and a test set of mildly and moderately-impaired patients. There is no overlap of patients between the training and test set.

E.2 KINEMATIC DATA DESCRIPTION

Each IMU captures 3-axis linear accelerations and angular velocities at 100 Hz. Angular velocities are converted to sensor-centric unit quaternions, representing the rotation of each sensor on its own axes, with coordinate transformation matrices. In addition, proprietary software (Myomotion, Noraxon) generates 22 anatomical angle values using a rigid-body skeletal model scaled to patient height. See Section E.6 for a detailed description of these angles. The resulting 76-dimensional vector thus represents the kinematic features of 3D linear accelerations, 3D quaternions, and joint angles from the upper body. As an additional feature, we included the paretic (stroke-affected) side of the patient (left or right) encoded in a one-hot vector, increasing the dimension of the feature vector to 77. Each entry (except paretic side) was mean-centered and normalized separately for each task repetition in order to remove spurious offsets introduced during sensor calibration.

E.3 VIDEO FEATURE EXTRACTION

We extracted and released features from raw videos using the X3D model (Feichtenhofer, 2020). The X3D model is a 3D convolutional network designed for performing the task of video classification, efficiently. The model is pretrained on the Kinetic dataset (Kay et al., 2017), which consists of coarse actions like running, climbing, sitting, etc. Since the *StrokeRehab* dataset consists of subtle, sub-second actions, we fine-tuned the X3D model on the training set of *StrokeRehab*. The fine-tuned model was then used to extract the frame-wise feature vectors from the raw videos.

E.4 DATA LABELING

An expert in the functional motion taxonomy (AP) (Schambra et al., 2019) individually trained the annotators, who underwent one month of intensive training on a series of increasingly complex activities. Once annotators labeled the training videos with high accuracy (less than 2% errors), they were given new videos to label independently. The annotators identified and labeled functional primitives in the video, which simultaneously labeled primitives in the IMU data. To ensure consistent labeling, the expert inspected one-third of all labeled videos. Interrater reliability between the

annotators and expert was high across primitives (Cohen's kappa: reaches, 0.96; repositions, 0.97; transports, 0.97; stabilizations, 0.98; idles, 0.96). One minute of recording took on average 73.9 minutes to annotate.

E.5 Description of the Rehabilitation Activities

Tables 8 and 9 describe the activities performed by the mildly and moderately impaired stroke patients in the cohort. Tables 10 and 11 describe the activities performed by the severely impaired patients assigned to Test set 2.

Activity	Workspace	Target object(s)	Instructions
Washing face	Sink with a small tub in it and two folded washcloths on either side of the countertop, 30 cm from edge closest to patient	Washcloths, faucet handle	Fill tub with water, dip washcloth on the right side into water, wring it, wiping each side of their face with wet washcloth, place it back on countertop. Use washcloth on the left side to dry face, place it back on countertop
Applying deodorant	Tabletop with deodorant placed at midline, 25 cm from edge closest to patient	Deodorant	Remove cap, twist base a few times, apply deodorant, replace cap, untwist the base, put deodorant on table
Hair combing	Tabletop with comb placed at midline, 25 cm from edge closest to patient	Comb	Pick up comb and comb both sides of head
Don/doffing glasses	Tabletop with glasses placed at midline, 25 cm from edge closest to patient	Glasses	Wear glasses, return hands to table, remove glasses and place on table
Eating	Table top with a standard-size paper plate (at midline, 2 cm from edge), utensils (3 cm from edge, 5 cm from either side of plate), a baggie with a slice of bread (25 cm from edge, 23 cm left of midline), and a margarine packet (32 cm from edge, 17 cm right of midline)	Fork, knife, re-sealable sandwich baggie, slice of bread, single-serve margarine container	Remove bread from plastic bag and put it on plate, open margarine pack and spread it on bread, cut bread into four pieces, cut off and eat a small bite-sized piece

Table 8: Description of the activities performed by the mildly and moderately impaired patients in the cohort (1/2).

E.6 Description of the Joint Angles

As described in Section 2.3, the sensor measurements are used to compute 22 anatomical angle values using a rigid-body skeletal model scaled to the patient's height and segment lengths. Table 12 describes these joint angles in detail.

Activity	Workspace	Target object(s)	Instructions
Drinking	Tabletop with water bottle and paper cup 18 cm to the left and right of midline, 25 cm from edge closest to patient	Water bottle (12 oz), paper cup	Open water bottle, pour water into cup, take a sip of water, place cup on table, and replace cap on bottle
Tooth brushing	Sink with toothpaste and toothbrush on either side of the countertop, 30 cm from edge closest to patient	Travel-sized toothpaste, toothbrush with built-up foam grip, faucet handle	Wet toothbrush, apply toothpaste to toothbrush, replace cap on toothpaste tube, brush teeth, rinse toothbrush and mouth, place toothbrush back on countertop
Moving object on a horizontal surface	Horizontal circular array (48.5 cm diameter) of 8 targets (5 cm diameter)	Toilet paper roll	Move the roll between the center and each outer target, resting between each motion and at the end
Moving object on/off a Shelf	Shelf with two levels (33 cm and 53 cm) with 3 targets on both levels (22.5 cm, 45 cm, and 67.5 cm away from the left-most edge)	Toilet paper roll	Move the roll between the center target and each target on the shelf, resting between each motion and at the end

Table 9: Description of the activities performed by the mildly and moderately impaired patients in the cohort (2/2).

Activity	Workspace	Target object(s)	Instructions *	
			Proximal >Distal	Proximal <Distal
Washing face	Sink with a small tub in it and two folded washcloths on either side of the countertop, 30 cm from edge closest to patient	Washcloths, faucet handle	Reach to touch faucet knob. Place washcloth in paretic hand and bring to both sides of face.	Open and close faucet. Lift washcloth from basin and wring it out.
Applying deodorant	Tabletop with deodorant placed at midline, 25 cm from edge closest to patient	Deodorant	Reach to touch deodorant. Place deodorant in paretic hand and bring to opposite armpit.	Lift deodorant for 3 seconds. From the horizontal position, rotate deodorant upright and return to original position.
Hair combing	Tabletop with comb placed at midline, 25 cm from edge closest to patient	Comb	Reach to touch comb. Place comb in paretic hand and bring to both sides of head.	Lift comb for 3 seconds.
Don/doffing glasses	Tabletop with glasses placed at midline, 25 cm from edge closest to patient	Glasses	Reach to touch glasses.	Lift glasses for 3 seconds.
Eating	Table top with a standard-size paper plate (at midline, 2 cm from edge), utensils (3 cm from edge, 5 cm from either side of plate), a baggie with a slice of bread (25 cm from edge, 23 cm left of midline), and a margarine packet (32 cm from edge, 17 cm right of midline)	Fork, knife, re-sealable sandwich baggie, slice of bread, single-serve margarine container	Reach to touch each item separately on paretic side. Place fork in paretic hand and bring fork to mouth.	Lift each object on paretic side for 3 seconds.

Table 10: Description of the activities performed by the severely impaired patients in the cohort (1/2). * Instructions for the severely impaired patients were given based on the UE segment with greater preserved function. "Proximal > distal" indicates better strength in the proximal (i.e. deltoid, biceps, triceps) than distal (i.e. hand) UE, which was typically paralyzed in these patients. The initial UE position was generally at the edge of the table/counter closest to the patient. "Distal > proximal" had the opposite distribution of strength. The initial UE position was adjacent to the target object. All testing were done on the paretic UE.

Activity	Workspace	Target object(s)	Instructions*	
			Proximal >Distal	Proximal <Distal
Drinking	Tabletop with water bottle and paper cup 18 cm to the left and right of midline, 25 cm from edge closest to patient	Water bottle (12 oz), paper cup	Reach to touch object on paretic side. Reach across to touch object on non-paretic side.	Starting from upright position, lay object on paretic side horizontally, release, and return to upright. Perform same series of actions on the object on the non-paretic side.
Tooth brushing	Sink with toothpaste and toothbrush on either side of the countertop, 30 cm from edge closest to patient	Travel-sized toothpaste on left, toothbrush with built-up foam grip on right, faucet handle	Reach to touch object on paretic side. Place toothbrush in paretic hand and bring it to mouth.	Lift object on paretic side for 3 seconds.
Moving object on a horizontal surface	Horizontal circular array (48.5 cm diameter) of 8 targets (5 cm diameter)	Toilet paper roll (200 g) or can (200 g)	Investigator will assess if toilet paper roll can be grasped, or aluminum can if not. If grasp is possible, move roll/can between the center and each outer target, resting before and after each motion. If grasp is not possible, the toilet paper roll will be moved around the target array by the investigator and the patient will reach to touch it at each location.	Investigator will assess if toilet paper roll can be grasped, or aluminum can if not. If grasp is not possible, the toilet paper roll will be moved around the target array by the investigator and the patient will reach to touch it at each location.
Moving object on/off a Shelf	Shelf with two levels (33 cm and 53 cm) with 3 targets on both levels (22.5 cm, 45 cm, and 67.5 cm away from the left-most edge)	Toilet paper roll (200 g) or can (200 g)	Investigator will assess if toilet paper roll can be grasped, or aluminum can if not. If grasp is possible, move roll/can between the center and each outer target, resting before and after each motion. If grasp is not possible, the toilet paper roll will be moved around the target array by the investigator and the patient will reach to touch it at each location.	Investigator will assess if toilet paper roll can be grasped, or aluminum can if not. If grasp is not possible, the toilet paper roll will be moved around the target array by the investigator and the patient will reach to touch it at each location.

Table 11: Description of the activities performed by the severely impaired patients in the cohort (2/2).

Joint/segment	Anatomical angle
Shoulder	Shoulder flexion/extension Shoulder internal/external rotation Shoulder ad-/abduction Shoulder total flexion‡
Elbow	Elbow flexion/extension
Wrist	Wrist flexion/extension Forearm pronation/supination Wrist radial/ulnar deviation
Thorax	Thoracic* flexion/extension Thoracic* axial rotation Thoracic* lateral flexion/extension
Lumbar	Lumbar† flexion/extension Lumbar† axial rotation Lumbar† lateral flexion/extension

Table 12: List of anatomical angles. The system uses a rigid-body skeletal model to convert the IMU measurements into joint and segment angles. ‡ Shoulder total flexion is a combination of shoulder flexion/extension and shoulder ad-/abduction. *Thoracic angles are computed between the cervical vertebra and the thoracic vertebra. †Lumbar angles are computed between the thoracic vertebra and pelvis.