Characteristics of Visual Categorization of Long-Concatenated and Object-Directed Human Actions by a Multiple Spatio-Temporal Scales Recurrent Neural Network Model

Haanvid Lee **Minju Jung** **Jun Tani***
Department of Electrical Engineering
Korea Advanced Institute of Science and Technology
tani1216jp@gmail.com

Abstract

The current paper proposes a novel dynamic neural network model for categorization of complex human action visual patterns. The Multiple Spatio-Temporal Scales Recurrent Neural Network (MSTRNN) adds recurrent connectivity to a prior model, the Multiple Spatio-Temporal Scales Neural Network (MSTNN). By developing adequate recurrent contextual dynamics, the MSTRNN can learn to extract latent spatio-temporal structures from input image sequences more effectively than the MSTNN. Two experiments with the MSTRNN are detailed. The first experiment involves categorizing a set of human movement patterns consisting of sequences of action primitives. The MSTRNN is able to extract long-ranged correlations in video images better than the MSTNN. Time series analysis on neural activation values obtained from the recurrent structure shows that the MSTRNN accumulates extracted spatio-temporal features which discriminate action sequences. The second experiment requires that the model categorize a set of object-directed actions, and demonstrates that the MSTRNN can learn to extract structural relationships between actions and action-directed-objects (ADOs). Analysis of characteristics employed in categorizing both object-directed actions and pantomime actions indicates that the model network develops categorical memories by organizing relational structures between each action and appropriate ADO. Such relational structure may be necessary for categorizing human actions with an adequate ability to generalize.

1 Introduction

Recently, a convolutional neural network (CNN) [1], inspired by the mammalian visual cortex, has shown remarkably better object image recognition performance than conventional vision recognition schemes employing elaborately hand-coded visual features. A CNN trained with 1 million visual images from ImageNet [2] can classify hundreds of object images with an error rate of 6.67% [3], near-human performance [4]. However successful in the recognition of static visual images, CNNs have poor dynamic image processing capability. The model lacks the capacity to process temporal information. A typical approach to overcome this limitation has been to use a 3D Convolutional Neural Network (3D CNN) [5]. With this model, dynamic visual images can be recognized through convolutions in the temporal and spatial domains in a fixed window. 3D CNNs have shown good performance on many public human action video datasets such as UCF-101 [6] and HMDB-51 [7]. Even a CNN without any temporal processing capability is able to achieve recognition rates of 73% and 40.5% on UCF-101 and HMDB-51, respectively [8]. Baccouche et al. [9] has proposed a two-stage model to extract temporal information over entire image sequences, also adding a Long

Short-Term Memory (LSTM) network [10] as a second stage of a 3D CNN. Similarly, Venugopalan and colleagues [11] trained an LSTM (for the generation of corresponding descriptive sentences) extended CNN (for video processing) with nearly a hundred thousand videos and corresponding descriptions with good test results. And recently, Shi et al. [12] developed a convolutional LSTM that has LSTM cells embedded in the structure of the CNN for precipitation nowcasting.

For machine vision systems to learn the semantics of human actions, they have to perceive continuous visual streams and extract underlying spatio-temporal structures present in action patterns [13][14]. In the process, complex human actions can be characterized as compositions both in spatial and in temporal dimensions, which can then decomposed into and even creatively composed from reusable parts. "Temporal compositionality" represents goal-directed human actions as sequential combinations of commonly used behavior primitives [13]. "Spatial compositionality" represents the coordination of movement patterns in different limbs. Recognizing this distinction, Jung et al. proposed the Multiple Spatio-Temporal Scale Neural Network (MSTNN) [15] model to impose both spatial and temporal constraints on CNN neural dynamics. Specifically, Jung et al. adjusted the time constants of the lower layers to a smaller values than that of the higher layers in the MSTNN to make the temporal receptive field size of each layer to increase as the layer goes up. And the spatial receptive field size of each layer in the MSTNN also increases as the layer goes up as in the case of the CNN. As a result, the model could extract spatio-temporal features in a hierarchical manner. This characteristic of the MSTNN differentiates the MSTNN from the rest of the models that were discussed previously. And also, the characteristic is consistent with the neurophysiological evidences that increasingly large spatio-temporal receptive windows are observed in the human cortex [16][17]. In Jung et al.'s work, the MSTNN was able to categorize different sequential combinations of behavior primitives demonstrated by different subjects by learning from exemplar videos. However, the MSTNN is limited in that it extracts temporal features of input action videos only utilizing the slow damping dynamics of its leaky integrator neurons. Therefore, the stored information of extracted spatial features decays over time, leaving the model to depend largely on extracted spatial features of recent time steps instead. As a result, the model is not able to learn complicated, temporally extended sequences. In order to overcome these limitations, the current work adds leaky integrator neural units with recurrent connections at each level of the MSTNN. With this addition, extracted spatial features no longer decay over time. The extracted spatial features of previous time steps can be preserved, or decayed, or amplified with the recurrent weights. This new model, the Multiple Spatio-Temporal Scales Recurrent Neural Network (MSTRNN) contains both leaky integrator neural units without recurrent connectivity as feature units and leaky integrator neural units with recurrent connectivity as context units, with each type at each level employing identical time constants.

Learning compositional action sequences requires the development of temporal correlations in memory. The present work investigates how a recurrent neural structure can enhance this capacity, and compares the performance of the MSTRNN and MSTNN models in the categorization of long sequences of primitive actions to observe the degree of enhancement. Analysis of the internal representation of the MSTRNN further reveals how neural activation patterns develop inside the model. Then, the paper looks more deeply into the development of structural relationships between objects and transitive actions. For this purpose, a video dataset of object-directed human actions was prepared. This set of actions was designed so that the category of each action and action-directed object (ADO) could not be inferred in a trivial manner. Also, a pantomime (actions without ADOs) version of the dataset was prepared and the MSTRNN was tested on it to see if the model can infer correct ADOs from given pantomime action image sequences, including cases where non-ADOs (distractors) are present in image sequences. It is impossible for the model to perform correct ADO categorizations on the pantomime dataset without exploiting links between actions and ADOs that are learned from the training data of object-directed human actions.

2 Materials and Methods

The Multiple Spatio-Temporal Scales Recurrent Neural Network (MSTRNN) is a spatio-temporally hierarchical neural network model that classifies videos. The model has the structure of the Convolutional Neural Network (CNN) with additional recurrent structures included in each convolutional layer. The recurrent structure plays an important role in extracting latent temporal information from exemplar temporal sequences [18][19]. Context neurons constituting recurrent structures are leaky integrator neurons with time constants that are similar to those used in the Multiple Timescales Recurrent Neural Network (MTRNN) [20].

2.1 Model architecture

The MSTRNN model consists of the following four layers: input, context, fully-connected, and an output layer as shown in Figure 1 (A). The MSTRNN receives RGB image sequences in the input layer. Then from the image sequences, multiple context layers extract spatio-temporal features from the input image sequences. The extracted spatio-temporal features go through several layers of fully-connected layers. Then finally, in the output layer, the model classifies actions and action-directed-objects (ADOs) in object-directed human action videos. The output layer utilizes two softmax vectors to categorize both the action and the ADO of the input video.

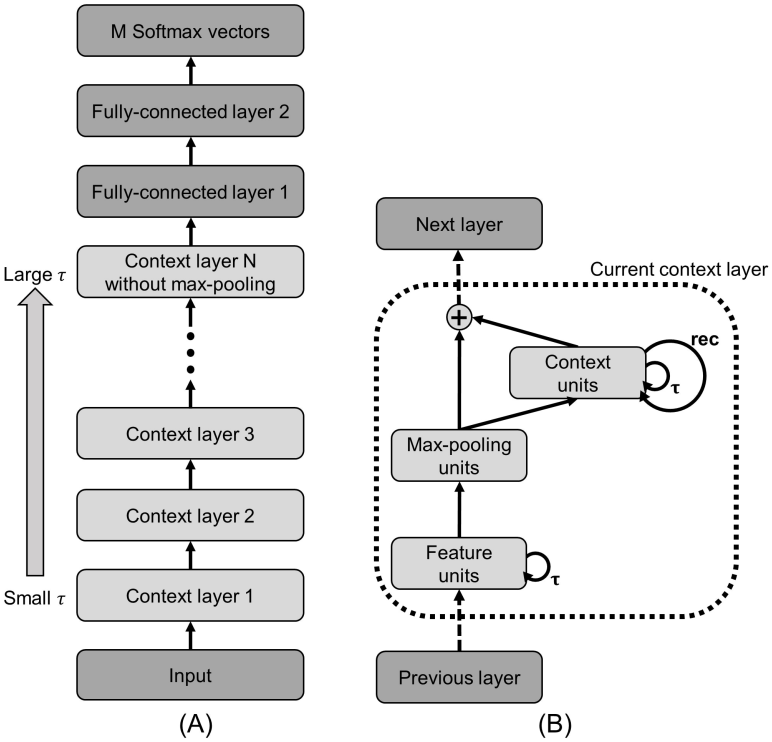

Figure 1: **The architecture of the MSTRNN.** (A) The full architecture of the MSTRNN. The MSTRNN consists of an input layer, *N* context layers, two fully-connected layers (optional), and *M* softmax vectors. *N* and *M* are adjustable numbers, and a τ is a time constant for a context layer. (B) The structure of the context layer. An arrow with a τ indicates decay dynamics of leaky integrator neurons in either the feature units or the context units. The arrow with "rec" indicates recurrent connections made on the context units

The context layer simultaneously extracts spatio-temporal features, and is the core building block of the MSTRNN. The context layer consists of feature units, pooling units, and context units as shown on Figure 1 (B). Each context layer is assigned a time constant that controls the decay dynamics of the context units and the feature units. A larger time constant makes the internal states of leaky integrator neurons in the context layer change more slowly at each time step. With a smaller time constant, the internal states of the leaky integrator neurons change faster at every time step. The MSTRNN assigns a larger time constant to higher layer leaky integrator neurons in order to develop a spatio-temporal hierarchy [15][20].

The feature units in a context layer are composed of leaky integrator neurons with time constants instead of static neurons as are normally used in a CNN [15]. The feature units are capable of extracting temporal features via decay dynamics of leaky integrator neurons, as well as capable of extracting spatial features by convolutional operations. Feature units extract features from context units and pooling units of the previous context layer as shown on Figure 1 (B). The features in the pooling units are extracted from them via convolutional kernels, and the features in the context units are extracted from them with the weights connecting the feature units and the context units belonging to the same retinotopic positions of the maps. Feature units and the context units in the same context layer have the same map sizes, and are connected in this way so that the feature locality is retained. The forward dynamics of the feature units are explained in Equation 1 and 2. The internal states and the activation values of the feature units at the *l*th context layer, the *m*th map of feature units, at time step *t* and at retinotopic coordinates (x, y) are represented as $\hat{f}^{txy}_{lm}$ and f^{txy}_{lm} in Equation 1 and 2, respectively.

$$\hat{f}^{txy}_{lm} = \Big(1 - \frac{1}{\tau_l}\Big)\hat{f}^{(t-1)xy}_{lm} + \frac{1}{\tau_l}\Big(\sum_{n=1}^{N_{l-1}} (k_{lmn} * p^{t}_{(l-1)n})^{xy} + b_{lm}\Big) + \frac{1}{\tau_l}\Big(\sum_{a=1}^{A_{l-1}} w^{xy}_{lma} c^{txy}_{(l-1)a}\Big) \quad (1)$$

$$f^{txy}_{lm} = 1.7159 \tanh\Big(\frac{2}{3}\hat{f}^{txy}_{lm}\Big) \quad (2)$$

where τ represents the time constant, *k* is the convolutional kernel, *b* is the bias used in the convolution operation, $*$ is the convolution operator, *N* is the total number of maps of the pooling units, *A* is the total number of maps of the context units. Additionally, *w* is the weight connecting the context units and the feature units, *p* and *c* are the activation values of the pooling units and the context units respectively. The first term on the right hand side of Equation 1 describes the decay dynamics of the leaky integrator neurons. The second term represents the convolution of the features in the pooling units. And, the third term describes the features extracted from the context units of the previous context layer. The hyperbolic tangent function recommended by LeCun et al. [21] is used as the activation functions of the feature units (Equation 2).

The context units consist of maps of leaky integrator neurons with recurrent weights. They extract spatio-temporal features in a similar manner to the feature units, except that they have enhanced temporal processing capacity by feeding back spatio-temporal information of the previous time step by the recurrent weights. To be specific, with recurrent connections, the context units exhibit recurrent dynamics in addition to decay dynamics normal to leaky integrator neurons. Therefore, the recurrent dynamics enhance the extraction of latent temporal features from input image sequences in the context units [18][19]. The recurrent connections are made to each neuron and also to the neurons of different maps in the same retinotopic positions to retain the locality of spatial features. Besides recurrent connections, context units have convolutional kernels that extract features from pooling units in the same layer (see Figure 1 (B)). The forward dynamics of context units are shown in Equation 3 and 4. Internal states and activation values for the *a*th map of the context units in the *l*th context layer at time step *t* and at retinotopic coordinates (x, y) are represented as $\hat{c}^{txy}_{la}$ and c^{txy}_{la} respectively.

$$\hat{c}^{txy}_{la} = \Big(1 - \frac{1}{\tau_l}\Big)\hat{c}^{(t-1)xy}_{la} + \frac{1}{\tau_l}\Big(\sum_{m=1}^{N_l} (\tilde{k}_{lam} * p^{t}_{lm})^{xy} + \tilde{b}_{la}\Big) + \frac{1}{\tau_l}\Big(\sum_{b=1}^{B_l} \tilde{w}^{xy}_{lab} c^{(t-1)xy}_{lb}\Big) \quad (3)$$

$$c^{txy}_{la} = 1.7159 \tanh\Big(\frac{2}{3}\hat{c}^{txy}_{la}\Big) \quad (4)$$

where τ represents the time constant, $\tilde{k}$ is the convolutional kernel, $\tilde{b}$ is the bias for the convolution operation, $*$ is the convolution operator, *N* is the total number of maps of the pooling units, *B* is the total number of maps of the context units, $\tilde{w}$ is the recurrent weight of the context units, *p* is the neural activations of pooling units. The first term on the right hand side of Equation 3 describes the decay dynamics of the leaky integrator neurons. The second term represents the convolution of the pooling units. The third term describes the recurrent dynamics in terms of recurrent weights. The neural activations of the context units in the previous time step are supplied through the recurrent weights. For the activation function of the context units, the model uses the same hyperbolic tangent function that was used for the feature units as shown in Equation 4.

2.2 Training method

Training was conducted in a supervised manner using the delay response scheme [15]. Black frames were input to the MSTRNN after each input image sequence during the delay response period. In this period, errors were calculated for each time step by comparing the outputs of the MSTRNN with the true labels of the input image sequences using Kullback-Leibler divergence. The cost function used in the training phase is shown in Equation 5. The error calculated for a whole input image sequence is represented as *E*.

$$E = \sum_{t=T+1}^{T+d} \sum_{n=1}^{N(s)} \sum_{s=1}^{S} \tilde{o}_{sn} \ln \left(\frac{\tilde{o}_{sn}}{o_{sn}^{t}} \right) \tag{5}$$

where *d* is the delay response period, *T* is the input video's duration (length of frames), *S* is the total number of softmax vectors in the output layer. *N(s)* is the total number of neurons in the *s*th softmax vector of the output layer, $\tilde{o}$ is the true output, and *o* is the output categorized by the MSTRNN. The true output was given as the one-hot-vector for each softmax vector. Here, the term one-hot-vector refers to a vector of values where a value is "1" for the one and only correct category and "0" for the rest of the categories. The error for each input action video is obtained with Equation 5. The error is used for the optimization of learnable parameters with the back propagation through time (BPTT) [22], and the stochastic gradient descent algorithms. The learnable parameters are, *k*, *b*, *w* of Equation 1, $\tilde{k}$, $\tilde{b}$, $\tilde{w}$ of Equation 3, weights held by the fully-connected layers, and the weights held by the output layer. To prevent overfitting, all learnable parameters (except biases) were learned with a weight decay of 0.0005 [23].

3 Results

The current study examines the performance of the Multiple Spatio-Temporal Scales Recurrent Neural Network (MSTRNN) given two types of experimental task. The first task compares the categorization performance of the model with that of the Multiple Spatio-Temporal Scales Neural Network (MSTNN) [15]. Then, the behavioral characteristics of the context layer in the MSTRNN is analyzed. The second task examines how the MSTRNN can learn to categorize a set of object-directed actions with an appropriate capacity to generalize.

3.1 Learning to categorize the 3 actions concatenated Weizmann dataset

3.1.1 The 3 actions concatenated Weizmann dataset

We compared learning and categorization capabilities of the MSTRNN and the MSTNN with a set of compositional long visual sequences. A set of exemplar video data was prepared by concatenating videos of 3 different human actions (jump-in-place (JP), one-hand-wave (OH), and two-hand-wave (TH) as shown on Figure 2) from the Weizmann dataset, resulting in 27 categories, and one video clip of concatenated actions for each category. 27 videos for each of 9 subjects exist in the dataset. The foreground silhouettes of the resulting 3 actions concatenated Weizmann (3ACW) dataset were emphasized by background subtraction utilizing background sequences, and were resized to 48x54 (48 pixels wide, 54 pixels high).

3.1.2 Experimental setting

The MSTRNN and the MSTNN models were trained (see the "Training method" section) with the same learning rate (0.1). Both models were trained for 50 epochs. For the evaluation of categorization performance, the leave-one-subject-out cross-validation (LOSOCV) scheme was used. On this method, for each of 50 training epochs, 1 subject was selected from the 9, his/her video clips were left out of the training data, and these were used as test data for that epoch. 9 sets of test accuracies from the 9 test subjects were averaged (rounded to the first decimal point). The highest accuracy is reported in evaluation of performance.

Structurally, both the MSTRNN and the MSTNN models have an input layer with one feature map for the input of grayscale video images. The size of the feature maps is 48x54. The models have identical output layer structures also, consisting of a softmax vector with 27 neurons. The models were designed to have only one softmax vector output since they only categorize the action category. Each of the softmax neurons in the vector represents one of 27 movement categories in the 3ACW dataset. Except for the input and output layers, the structure of the MSTRNN and the MSTNN are specified in Table 1 and 2, respectively. The only difference between their architectures is the addition of context units, and number of maps that consist the feature units in the layers of the models. The number of weights used in the MSTRNN and MSTNN were designed to be similar (MSTRNN: 495,327 weights, MSTNN: 497,506 weights) by adjusting the number of maps that are used in feature units in each layer of the MSTNN while also keeping similar ratio between the numbers of maps in adjacent layers (see Figure 1) (B)).

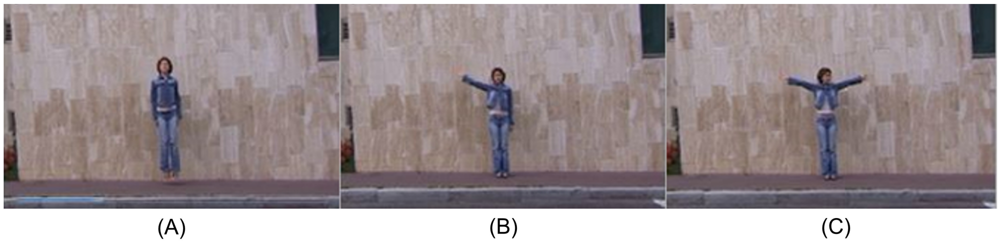

Figure 2: **The 3 human action categories used from the Weizmann dataset.** (A) Jump-in-place. (B) One-hand-wave. (C) Two-hand-wave

Table 1: **Parameters of the context layers in the MSTRNN model used in the experiment.**

		Feature units			**Poolingunits**			**Context units**		
Context layer	**Time constant**	**Kernel size**	**Map size**	**Total number**	**Pooling size**	**Map size**	**Total number**	**Kernel size**	**Map size**	**Total number**
1	2	5x11	44x44	6	2x2	22x22	6	7x7	16x16	3
2	5	7x7	16x16	50	2x2	8x8	50	8x8	1x1	25
3	100	8x8	1x1	100	-	1x1	-	1x1	1x1	50

Table 2: **Parameters of the context layers in the MSTNN model used in the experiment.**

			Feature units			**Pooling units**		
Layer	**Type**	**Time constant**	**Kernel size**	**Map size**	**Total number**	**Pooling size**	**Map size**	**Total number**
1	Convolutional	2	5x11	44x44	7	-	-	-
2	Max-pooling	-	-	-	-	2x2	22x22	7
3	Convolutional	5	7x7	16x16	58	-	-	-
4	Max-pooling	-	-	-	-	2x2	8x8	58
5	Convolutional	100	8x8	1x1	116	-	-	-

3.1.3 Experimental results

The categorization accuracy on the 3ACW dataset was 83.5% for the MSTRNN and 44% for the MSTNN (as shown in Figure 3). This result implies that context units with recurrent weights improve categorization of long concatenated human action sequences.

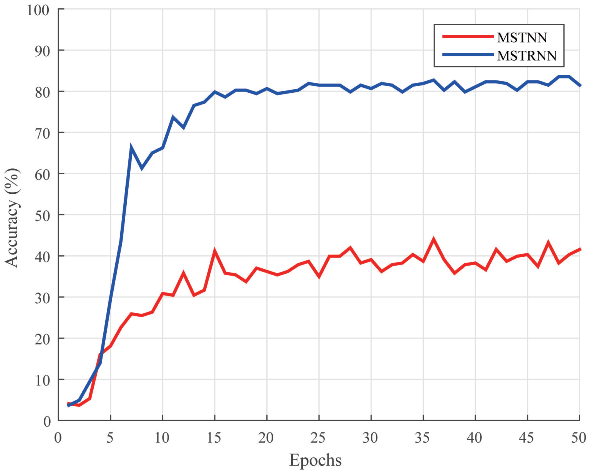

Figure 3: **The mean categorization accuracy on the 3ACW dataset along the training epochs.**

Internal dynamics during the ideal case, when all videos of a test subject left out from the training of the model were classified correctly, were assessed by time series analysis on the neural activation values obtained from the last context layer of the MSTRNN. The time series of neural activations were visualized by using principle component analysis (PCA) [24]. Only the first and the second principle components of the neural activations were used for the visualization. In the following discussion of this analysis, “X” indicates arbitrary primitive actions, and the time series of neural activations obtained when given action primitives A, B, C in a sequential manner is indicated by trajectory A-B-C.

Neural activations of the feature units (Figure 4) at any given time step are largely affected by the primitive action of the input video at that time step. Accordingly, neural activations approached points representative of the current action in the PCA mapped space. Trajectories JP-X-X, OH-X-X, TH-X-X approached markers “JP”, “OH”, “TH” respectively before the presentation of second primitives, as shown in Figure 4 (A). Figure 4 (B) shows trajectories of X-JP-JP in order to illustrate feature unit characteristics more clearly. In this figure, the JP-JP-JP trajectory approached the marker “JP” in the figure. But the trajectory OH-JP-JP and TH-JP-JP first approached regions marked “OH” and “TH” respectively, before the second and third primitives were shown. After the second and the third primitives (JP and JP), the trajectories changed their paths and approached the region marked “JP”. Similar characteristics are observed in JP-JP-X trajectories, as well (see Figure 4 (C)).

The decay dynamics of the feature units are responsible for trajectories in Figure 4 approaching the markers that represent current action primitives. As described in Equation 1, the internal neural values of the feature units are affected by both current spatial-temproal features processed from the previous context layer and the internal neural values of the units of the previous time step. On this method, previously extracted spatial features does not effectively affect the internal neural values of the current time step to keep the track of which action primitives came in the past since they gradually decay over time. This makes the current trajectories approach abstract points (markers “JP”, “OH”, “TH” in Figure 4) in PCA mapped space corresponding to the primitive actions with which the model is presented.

Unlike the trajectories obtained from the feature units, the trajectories obtained from the context units of the last context layer do not simply approach positions representative of currently displayed action primitives in the PCA mapped space, but tend to differentiate the primitives from each other. Due to the recurrent structure in the context units, the memory of the context units does not simply decay as in feature units. Rather, the units retain important spatio-temporal features extracted during previous time steps and more or less reinforced during current steps. In this way, by accumulating extracted spatio-temporal features over time during training, the trajectories obtained from context units is able to differentiate different primitives shown to the MSTRNN during testing. As shown in Figure 5 (A), the trajectories that start with same first primitives passes through same path before the second primitives are shown to the MSTRNN. The trajectories differentiates in terms of the second primitives when they appear (Figure 5 (B)). And, the trajectories are further differentiated in terms of the third primitives as well (Figure 5 (C)). The differences between the trajectories of context units and feature units come clear in their direct comparison (as shown in Figure 5 (D) and Figure 4 (B) for X-JP-JP trajectories).

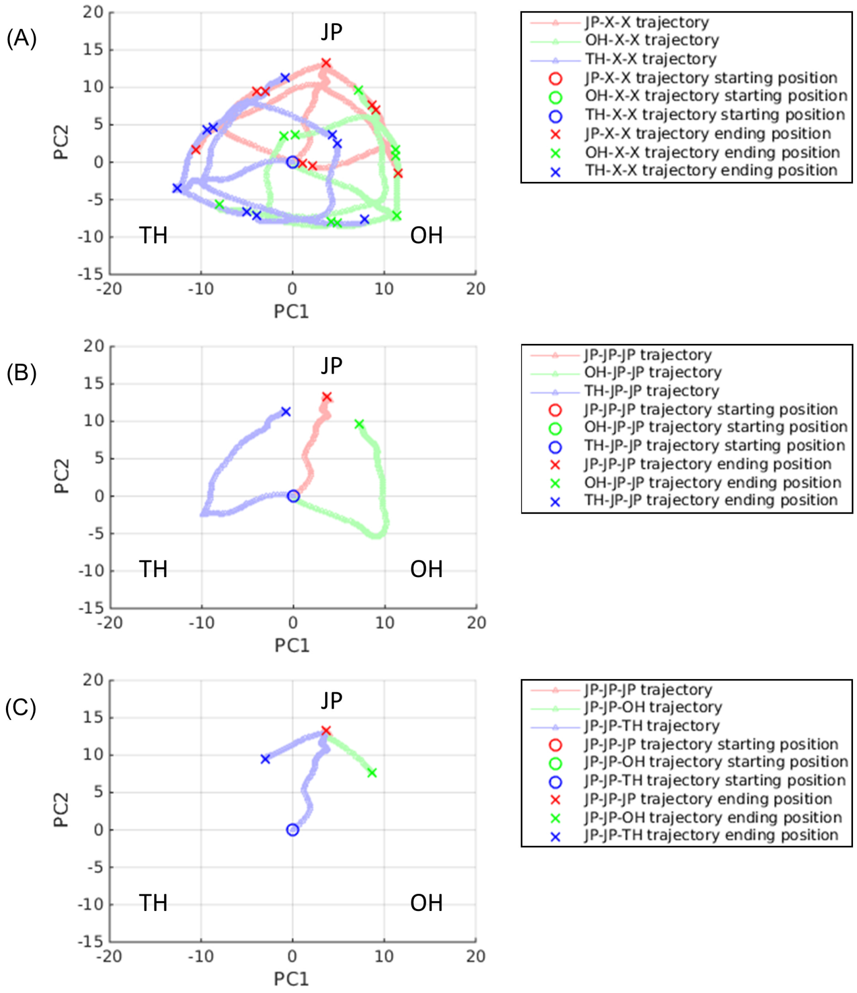

Figure 4: **Time series neural activation patterns of the feature units in the third (last) context layer of the MSTRNN as analyzed by PCA.** First and second principle components of the activation patterns were used to visualize the activation patterns. (A) Time series neural activation values obtained by feeding the MSTRNN input videos of all categories. (B) Time series neural activation values obtained by feeding the MSTRNN input videos with the same second and third primitive actions. (C) Time series neural activation values obtained by feeding the MSTRNN input videos with the same first and second primitive actions.

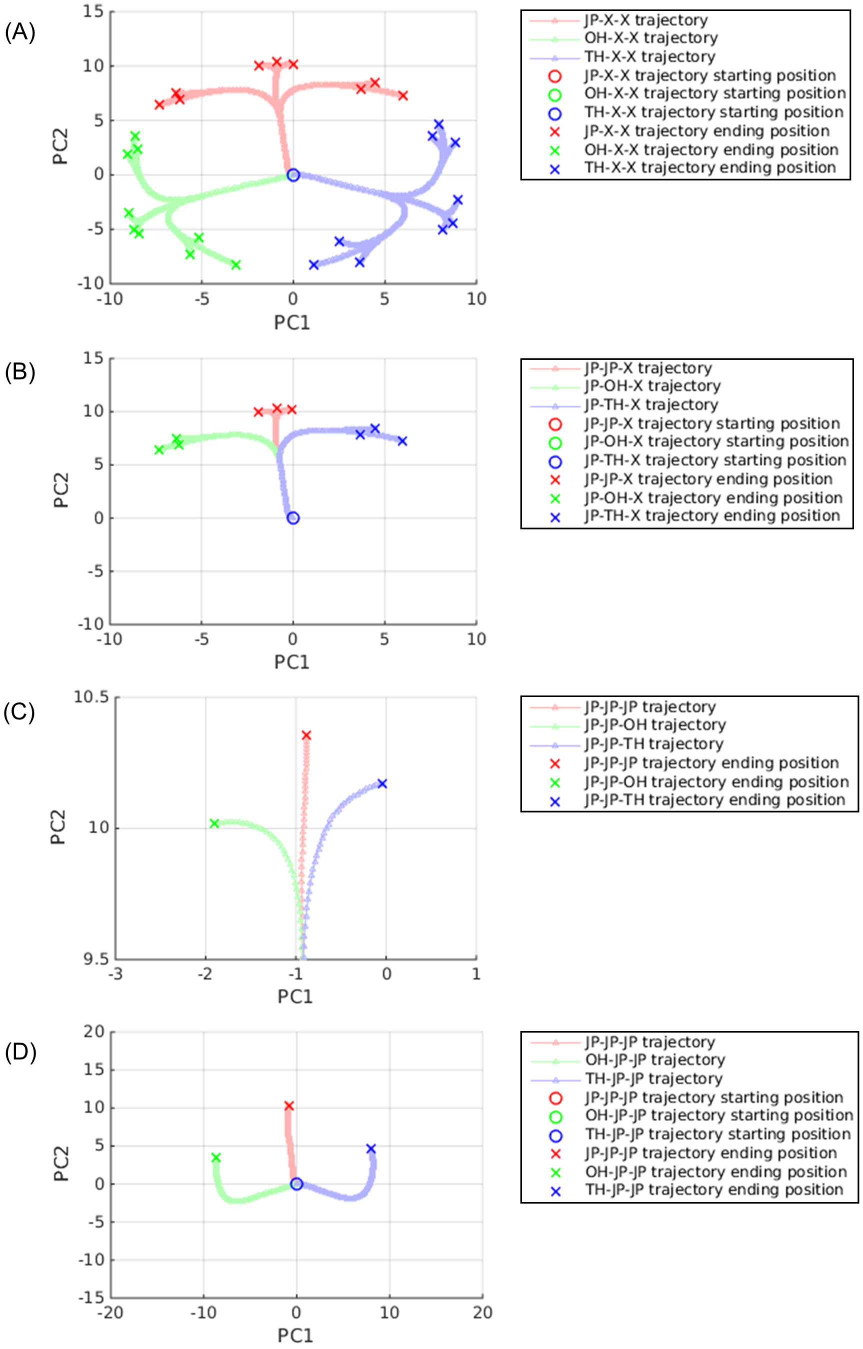

Figure 5: **Time series neural activation patterns of the context units in the third (last) context layer of the MSTRNN as analyzed by PCA.** The first and second principle components of the activation patterns were used to visualize the activation patterns. (A) Time series neural activation values obtained by feeding the MSTRNN input videos of all categories. (B) Time series neural activation values obtained by feeding the MSTRNN input videos with the same first primitive (JP) actions. (C) Time series neural activation values obtained by feeding the MSTRNN input videos with the same first and second primitive actions (JP, JP). (D) Time series neural activation values obtained by feeding the MSTRNN input videos with the same second and third primitive actions (JP, JP).

3.2 Learning to categorize the object-directed human action dataset

This experiment examines how accurately the MSTRNN is able to categorize actions and corresponding action-directed-objects (ADOs) as represented in object-directed human action videos, also answering if the MSTRNN can distinguish ADOs from both non-ADOs and ADOs present in image sequences by learning structural links between the actions and the ADOs. To this end, the MSTRNN was tested on a pantomime version (actions without ADOs) of the object-directed human action dataset (ODHAD), both with and without non-ADOs present as distractors. Performance was compared with the result obtained when the MSTRNN was presented with objects-present test data to confirm that the MSTRNN does develop such structural links.

3.2.1 Object-directed human action dataset

The first experiment involves the categorization of object-directed actions, and for this, an object-directed human action dataset (ODHAD) consisting of 9 subjects performing 9 actions directed at 4 objects was prepared. There were 15 object-directed action categories in total (as shown in Figure 6 and Table 3). The dataset was designed so that the categorization task is non-trivial. We introduced a non-ADO as a distractor along with an ADO in each video scene to prevent the model from inferring a human action or an ADO solely through recognizing the presence of a certain object in a video. For each object-directed action category, 6 videos were shot for each subject with three different non-ADOs appearing in two videos each in different states (opened or closed) except the cup (which does not have more than one state) as they are presented in the other videos as ADOs. During dataset recording, subjects generated each action naturally and without constraints on movement trajectory or speed. For manipulating objects, left-handed subjects were free to use their left hands. Objects were located in various positions in the task space. However, the camera view angle was fixed (the problem of view invariance is beyond the scope of the current study).

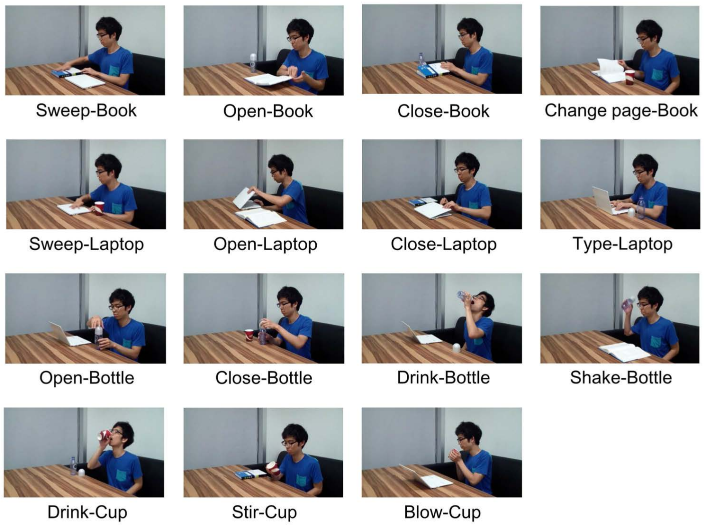

Figure 6: **A sample frame for each object-directed action from the object-directed human action dataset.** A distractor object appears along with an action-directed object in each video of the dataset. Distractors except the cup (book, laptop, and bottle) appears in videos in all possible states (opened or closed).

Table 3: **Composition of the ODHAD.**

	Actions								
ADOs	Sweep	Open	Close	Drink	Change-page	Type	Shake	Stir	Blow
Book	1	2	3	-	4	-	-	-	-
Laptop	5	6	7	-	-	8	-	-	-
Bottle	-	9	10	11	-	-	12	-	-
Cup	-	-	-	13	-	-	-	14	15

The second experiment in categorization of object-directed actions required preparation of two pantomime versions of the ODHAD. The pantomime version of ODHAD refers to the videos of human actions without ADOs. Although the actions look like as if the subjects are using imaginary objects in their actions. One pantomime version of the dataset had no objects in the scene, and the other included only distractors (non-ADOs) in the scene. Both versions of the dataset were prepared for the joint categories of Drink-Cup and Stir-Cup, as was the original version of the dataset for these two joint categories only compiled from 5 subjects whose video data had not been used during training with the ODHAD in the first experiment.

3.2.2 Experimental setting

The MSTRNN model used for the experiments with the ODHAD and its pantomime versions has one input layer (three 110x110 feature maps), four context layers, two fully-connected layers (512 dimensions), and an output layer with two softmax vectors to indicate categorized objects (a softmax vector with 4 neurons) and actions (a softmax vector with 9 neurons). Table 4 shows the parameter settings of the context layers in the MSTRNN model. The time constants of the context layers were picked both manually and heuristically, and it was observed that the MSTRNN performed best when small, large time constants were assigned to the lower, higher layers, respectively. Time constants of 2, 3, 5, and 110 were used for the first to the fourth context layers, respectively. The learning rate of the model started at 0.01 and decayed by 2% every epoch. The model network was trained for 130 epochs. In addition to using weight decay to alleviate the risk of overfitting (see the "Training method" section), the dropout technique was applied to the fully-connected layers of the MSTRNN with a dropout rate of 70% [25]. Also to alleviate the overfitting problem, 100x100 pixel patches were randomly sampled from the 110x110 pixel images, and were used as a training data to make the recognition of object-directed human actions more robust to locational translations of input images [26].

Table 4: **Parameter settings of the context layers in the MSTRNN.**

		Feature units			**Pooling units**			**Context units**		
Context layer	**Time constant**	**Kernel size**	**Map size**	**Total number**	**Pooling size**	**Map size**	**Total number**	**Kernel size**	**Map size**	**Total number**
1	2	5x5	96x96	20	2x2	48x48	20	5x5	44x44	20
2	3	5x5	44x44	50	2x2	22x22	50	5x5	18x18	50
3	5	5x5	18x18	90	2x2	9x9	90	4x4	6x6	90
4	110	4x4	6x6	230	2x2	3x3	230	3x3	1x1	230

The leave-one-subject-out cross-validation (LOSOCV) scheme was used to assess overall categorization accuracy in a similar manner to the way it was used for the experiment with the 3ACW dataset. The dataset excluding one subject's data was used for training the MSTRNN, and the excluded subject's data was used for testing at each epoch, resulting in accuracy measures for 130 epochs. As there are 9 ways of excluding one subject's data for the purpose of testing among 9 subjects, there are 9 sets of test accuracies, each set containing accuracies for all 130 epochs. By averaging these 9 sets of test accuracies, we obtained an averaged set of test accuracies for 130 epochs. Among the averaged set of accuracies, the epoch that showed the maximum recognition accuracy on the joint category of action-ADO pair was found. The averaged categorization accuracies of this epoch on the action, the ADO, and the joint category of action-ADO pair were rounded off to the first decimal place and recorded as measures of overall performance.

In the experiment with the two pantomime datasets, the MSTRNN used the set of parameters learned from the first experiment conducted with the object-directed human action dataset (ODHAD). The set of parameters is the one that was trained with ODHAD until the epoch where the model showed the maximum recognition accuracy on the joint category of action-ADO pair. The learned parameters were tested on the ODHAD and the two pantomime datasets, each taken from 5 test subjects, and containing the joint categories of Drink-Cup and Stir-Cup. The recognition accuracy of the MSTRNN on the joint action-ADO category was measured and rounded off to the first decimal place.

3.2.3 Experimental results

For the first experiment in categorization of object-directed actions, the overall categorization accuracies for ADOs, actions, and action-ADO pairs were 81.9%, 75.9%, and 68.9% respectively. From these average recognition accuracies, it can be seen that - although the MSTRNN model used in the experiment can categorize object-directed action patterns in the test data with generalization to some degree - the model exhibited a certain amount of mis-categorization. See the video in the link (https://sites.google.com/site/mstrnn1/) for demonstration of the MSTRNN categorizing test videos of the ODHAD.

Next, we examined structural links developed between actions and ADOs. Table 5 shows ADO categorization rates during test conditions where specific ADOs and non-ADOs were present in the test data. Where the MSTRNN failed to correctly categorize an ADO, it was most likely to confuse present non-ADOs as ADOs in 9 out of 12 test conditions (see Table 5). This result indicates that the model network was successful in capturing structural links between objects and actions directed at them, although exhibiting a tendency to be more or less distracted by present non-ADOs more or less similar to the ADO, i.e. the network mistook a book for a laptop and a laptop for a book in 19.4% and 18.1% of cases, respectively.

Table 5: **ADO categorization rate on all possible cases of ADO, DO pair present in the input videos.**

Objects in the videos		Categorized ADOs (%)			
ADOs	**Non-ADOs**	**Book**	**Laptop**	**Bottle**	**Cup**
Book	**Laptop**	76.4	19.4	0	4.2
Book	**Bottle**	80.6	4.2	8.3	6.9
Book	**Cup**	81.9	4.2	4.2	9.7
Laptop	**Book**	18.1	79.2	1.4	1.4
Laptop	**Bottle**	0	87.5	5.6	6.9
Laptop	**Cup**	1.4	84.7	6.9	6.9
Bottle	**Book**	1.4	4.2	84.7	9.7
Bottle	**Laptop**	1.4	12.5	84.7	1.4
Bottle	**Cup**	0	0	81.9	18.1
Cup	**Book**	11.1	0	9.3	79.6
Cup	**Laptop**	1.9	5.6	13	79.6
Cup	**Bottle**	0	0	13	87

The ADO categorization rates were rounded off to the first decimal place.

After categorization of all object-directed action cases, neural activity in the higher layer was analyzed. Neural activation values of the last time step in the second fully connected layer were plotted in 2-dimensional space by principle component analysis (PCA) [24]. Input image sequences from which the MSTRNN generates similar ouputs are identifiable by the clusters of neural activations in the PCA mapping shown in Figure 7. Neural activations first cluster by the ADO category (markers of same colors) and by the action category (markers of same shapes). Also, the PCA mapping can be interpretted as having two large clusters of book and laptop; bottle and cup ADO pairs (marked by symbols A and B in Figure 7). Then, those groups of clustered points make sub-clusters according to action category. Finally, these groups sub-cluster according to action-ADO joint category (markers of both same colors and shapes).

Figure 7 shows that neural activations of the same action-ADO pairs cluster more closely together than do clusters of other categories of neural activations. It is interesting to note that input image sequences with the same action-ADO pairs are not mapped to exactly the same postions. This is because, even if the test videos belong to the same joint category, present non-ADOs, ADO tragectory and orientation, test subject idiosyncrasies and many other variables all differ. The next most closely gathered neural activations in general are the ones that have same action categories. Action-ADO clusters of neural activations can be viewed as subgroups of a class of activations that have the same actions in general, indicated in Figure 7 with same marker shapes. For example, though they employ different objects, Open-Book and Open-Laptop are located close together in the PCA mapped space as they have the same action category. Compare this result with that for Open-Bottle. Though still appearing on the same side of the map, and closer to Open-Book or Open-Laptop than to other action-ADO pairs, Open-Bottle neural activations appear relatively farther away because image sequences for Open-Bottle differ from Open-Laptop or Open-Book more than these do from each other.

Neural activations also group according to ADO. Figure 7 shows that neural activations generated from test videos of the same joint categories organize into overgroups according to the four ADO categories. However, these groups are distributed differently than are the groups for actions. In Figure 7, neural activations with the same ADO are more spread out than are activations with the same actions. For example, neural activations of bottle and cup ADO categories are spread out while neural activations of Drink-Bottle and Drink-Cup are tightly clustered. Finally, neural activations cluster by book and laptop, bottle and cup ADO pairs. Figure 7 indicates the neural activations of book and laptop, and bottle and cup ADO pairs with red and blue line borders. For example, book and laptop neural activations – as a pair - appear far from those of bottle and cup neural activations - as a pair – even with same actions (such as "open" and "close") since representative image sequences differ a great deal more between these pairings than within them. By analyzing neural activity according to these ADO pairs, we also see that same-action neural activations with different ADOs subcluster at the boundaries where relevant ADO cluster regions meet. For example, Drink-Bottle and Drink-Cup appear relatively close together, especially given the distance separating other Cup from Bottle activations. Similar cases can be observed in neural activations generated by the book and the laptop test data as well, for example Sweep-Book and Sweep-Laptop, Open-Book and Open-Laptop, Close-Book and Close Laptop – all of these form subclusters of activity.

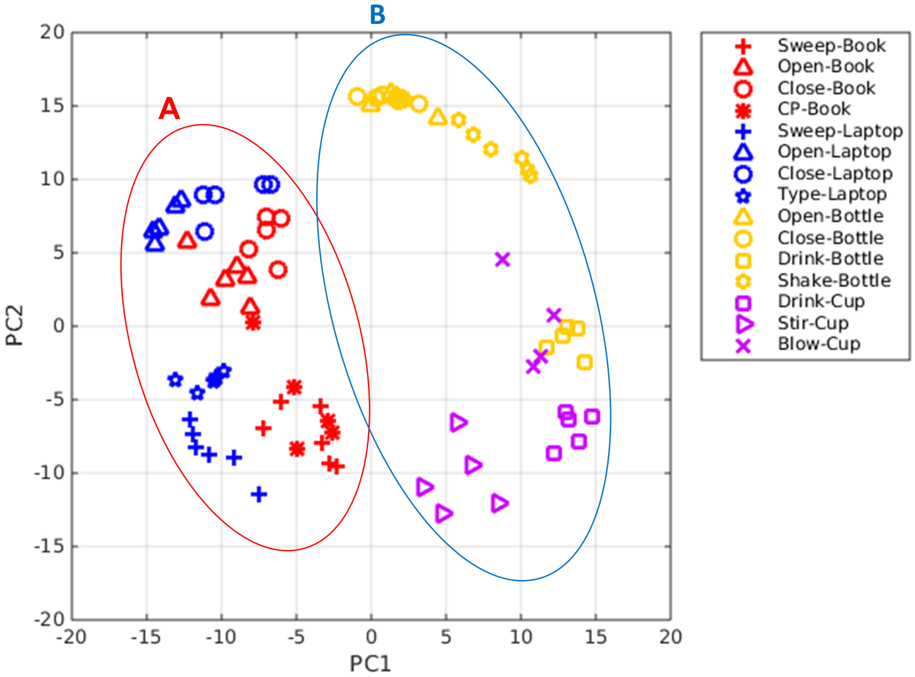

Figure 7: **PCA mapping of activation values generated from the second fully-connected layer of the MSTRNN given object-directed human action test data.** First and Second principle components were used for the visualization. The neural activations of correctly categorized test videos can be largely divided in to two groups according to the two ADO pairs: book and laptop (symbol A), bottle and cup (symbol B). The two groups have subclusters according to action categories (same shape). And the clusters of action category can be further divided into subclusters of neural activations that have same joint categories.

When the MSTRNN categorizes an action category of an input video correctly, the possible number of corresponding ADOs is (in most cases) two. For example, if the model categorizes an action category of drink correctly, then the possible corresponding ADO is either bottle or cup. On the other hand, with the ADO correctly categorized, there are more possible corresponding actions. For example, when the ADO is correctly categorized as laptop, there are four possible actions: sweep, open, close, change-page (Figure 7). Interestingly, as evident in the PCA mapping, the MSTRNN exhibits higher recognition accuracy for the ADO category (81.9%) than the action category (75.9%). And, in the clustering structure illustrated in Figure 7, it can be seen that the MSTRNN categorizes ADOs prior to actions. Here we see that the clusters of neural activations generated from the same actions are subclusters of larger book and laptop, bottle and cup groups. In the end, what is strongly evident from this preceding analysis of clustering structures evidenced in the PCA mapping of MSTRNN neural activations is that the model learns structural links between actions and corresponding ADOs from training data, with these relationships then facilitating ongoing agency within the given action-ADO environment.

Having established that the MSTRNN is able to learn structural links between actions and ADOs, we investigated if the MSTRNN is capable of inferring correct ADOs when ADOs are absent from test image sequences, and even when only non-ADO (distractors) appear, instead. Table 6 compares action-ADO categorization rates given Drink-Cup and Stir-Cup test data with rates given pantomime version test of data both with and without distractors (note that Table 6 includes only joint categories with categorization accuracies greater than 0%).

Table 6: **Action-ADO (joint category) categorization rates using object-directed test videos and their pantomime version test data with and without distractors present in the scene.**

	Categorized joint categories (%)								
Actual input	**Sweep -Book**	**Sweep -Laptop**	**Close -Bottle**	**Drink -Bottle**	**Shake -Bottle**	**Sweep -Cup**	**Drink -Cup**	**Shake -Cup**	**Stir -Cup**
Drink-Cup	0	0	0	16.7	3.3	0	80	0	0
Stir-Cup	3.3	3.3	0	0	0	3.3	0	0	90
Drink-Cup pantomime	0	0	0	6.7	0	0	93.3	0	0
Stir-Cup pantomime	0	0	0	0	6.7	0	3.3	0	90
Drink-Cup pantomime with distractors	0	0	0	30	0	0	70	0	0
Stir-Cup pantomime with distractors	3.3	0	3.3	13.3	30	0	6.7	3.3	40

The joint category categorization rates were rounded off the the first decimal place.

Action-ADO recognition accuracies are slightly better with the pantomime test videos than with the original object-directed human action test videos. For example, Table 6 shows that the model correctly categorized the Drink-Cup pantomime correctly in 93.3% of instances, and incorrectly as Drink-Bottle in 6.7%. It is worth noting that the pantomime actions of Drink-Cup and Drink-Bottle are quite similar, so that even human beings may mis-categorize these pantomimed actions. Consider also the categorization accuracy using the Stir-Cup pantomime test videos without distractors. Performance here is similar to accuracies obtained using the object-present Stir-Cup test data.

When given distractors-present pantomime test data, the MSTRNN demonstrated a tendency to be distracted by non-ADOs. Table 6 shows that the MSTRNN recognized Drink-Cup pantomime action videos with distractor non-ADOs present as Drink-Cup (70%) and as Drink-Bottle (30%). But again, pantomimed Drink-Cup and Drink-Bottle are difficult even for humans to distinguish, and so such ambiguous cases may be discounted. Performance was worst with the Stir-Cup pantomime test data with distractors. The action-ADO category of the test videos were mis-categorized more than the other test conditions (Stir-Cup, Stir-Cup pantomime in Table 6) by 50%. This significant amount of mis-categorization is caused by the MSTRNN mis-categorizing distractors as ADOs. Because, the categorization performance of the MSTRNN on the Drink-Cup pantomime and Stir-Cup pantomime test videos without distractors were similar or better than its performance on the cup and non-ADOs present Drink-Cup, Stir-Cup test videos (Table 6). Indeed, the MSTRNN correctly categorized ADO (cup) of the test videos by only 50% which is lower than the ADO categorization accuracies of the other test conditions (Stir-Cup: 93.3%, Stir-Cup pantomime: 93.3%) in Table 6. In the end, the preceding analysis shows that the ADO categorization process of the MSTRNN depends on both currently perceived objects and actions.

4 Discussion

In the first experiment categorizing long-ranged videos of compositional human action sequences, the proposed Multiple Spatio-Temporal Scales Recurrent Neural Network (MSTRNN) model performed better than the previous Multiple Spatio-Temporal Scales Neural Network (MSTNN) [15] model. The recurrent connectivity in the context units enhanced the capability of the model to extract long-term correlations latent in training data. Analysis of the internal dynamics of the context layer demonstrated that the feature units in the context layer tended to capture spatio-temporal features of recently given input images over a short time interval. On the other hand, the context units in the context layer captured spatio-temporal features of image sequences over a relatively longer period of time due to its recurrent structure and larger time constant.

The second experiment tasked the MSTRNN with learning to categorize object-directed human actions in order to examine the model's ability to capture underlying spatio-temporal structures linking human actions and visual images of objects at which the actions are directed. The overall categorization accuracies on the action category and the action-directed-object (ADO) category were 75.9% and 81.9%, respectively. These categorization results demonstrate that the MSTRNN is capable of learning structural links between actions and corresponding ADOs by extracting spatio-temporal features in its context layers. The MSTRNN was distracted by non-ADOs present in image sequences, but could successfully recognize ADOs in most cases. Principle component analysis (PCA) [24] of the neural activations of the last time step in the second fully-connected layer shows that the model developed structural links between actions and corresponding ADOs. This is evident in the PCA mapping as similar neural activations are mapped to similar locations in the PCA mapped space (as shown in Figure 7). In PCA space, neural activations generated from test data with the same joint category are more similar than neural activations generated from test data with same action, and both are more similar than neural activations generated from test data with merely the same ADO.

Testing reveals that the MSTRNN can infer ADOs from pantomime action videos by exploiting the structural links between actions and ADOs learned in training. However, the MSTRNN demonstrated a tendency to be distracted when tested on the pantomime action videos with non-ADO distractors present, and its ADO categorization accuracies were lower in these cases than with pantomime test videos without distractors present. These results imply that the MSTRNN depends on spatio-temporal features extracted from sequences of action images as well as on static object presence in order to correlate a given action with its appropriate ADO.

Currently, the best action-ADO recognition rate obtained in the second experiment is 68.9%, which should be improved in future study. One of the main reasons for such a significant degree of mis-categorization might be overfitting. Overfitting may be alleviated with recently developed deep learning regularization techniques including the dropout technique for recurrent connections [27]. By applying the dropout technique to Long Short-Term Memory (LSTM), Gal et al. came up with the Variational LSTM which has demonstrated less susceptibility to problems of overfitting than the standard LSTM. Recurrent batch normalization [28] may also help to alleviate overfitting. Cooijmans et al. have shown that batch-normalization of LSTM improves its generalization capacity and encourages faster convergence in the learning phase.

Good performance depends on appropriate model parameters. In the current study, it was found that the performance of the model depends crucially on time constant values ascribed to each context layer. Since there is no analytical way to determine the optimal values for time constants, these must be established heuristically. Future study should investigate a scheme for the self-adjustment of time constants. Two approaches immediately present themselves. One involves using LSTM [10] or Gated Recurrent Units (GRU) [29] to adapt time constants at each time step. LSTM recurrent neural networks have demonstrated outstanding performance on sequence-based tasks with long-term dependencies [30]. And, the recently developed GRU has demonstrated similar or higher performance [31]. Embedding these models in the structure of our proposed model will make a model that is somewhat similar to the convolutional LSTM [12]. From the results that were reported so far, the LSTM and the GRU do not develop hierarchical structures while learning from data, so constraints on time-constant adaptation must be applied to each layer. Another approach to self-adapting time constants involves their automatic determination by way of genetic algorithm. On this approach, simulated robot experiments demonstrate that the network naturally develops slow dynamics in the higher layer [32]. Therefore, integration with the current work seems promising.

Future study should also investigate the possibility of improving categorization performance by modifying the structure of the MSTRNN. First, by implementing a top-down prediction and attention pathway in addition to the current bottom-up pathway, MSTRNN action recognition performance may improve because top-down processes provide values for anticipated future perceptual events. Also, a recurrent loop may be added from the higher layer to the lower layer. Currently, recurrent connections are made only within each context layer, and as a consequence, extracted spatio-temporal information in the higher layer cannot affect lower layer neural activations. Recurrent structures connecting higher to lower layers should facilitate this influence, and as a result, the capability of the MSTRNN to extract latent features of sptaio-temporal patterns may be enhanced.

Ongoing work is focusing on improving the MSTRNN to enhance the development of structural links via learning and involves making a dataset to be tested on an improved version of the MSTRNN.

The dataset will be more complex in terms of action-ADO compositionality, requiring the use of transitive verbs, object nouns and modifiers, e.g. PUT-CUP-ON-BOOK. The improved version of the MSTRNN will be tested on this new dataset to see if the model is able to learn more complex structural links of transitive verbs, object nouns, and modifiers.

5 Conclusion

Biological evidence exists that humans learn temporal sequences through recurrent neural networks in their brains. Goel et al. have shown not only that recurrent neural circuits and cortical circuits in particular are capable of encoding time, but also that there are mechanisms in place that allow such circuits to 'learn' the temporal structures of stimuli [33]. Also, recent neurophysiological studies suggest that human object recognition is aided by the understanding of object relevant actions [34]. As a recurrent neural network model for object-directed human action recognition, the MSTRNN is inspired by such biological insights. The MSTRNN also recognizes action-directed-objects (ADOs) in light of perceived actions, and has demonstrated a capacity to infer correct ADOs from pantomime action videos. This capacity is grounded on learned structural links between actions and corresponding ADOs, a capacity to be emphasized in future work.

In summary, the MSTRNN model network developed categorical memories for a set of trained object-directed actions, self-organizing an internally consistent relational structure among them within a bottom-up generated metric space. Development of such a relational structure in metric space may facilitate generalization in categorization. But, a fundamental question presents itself: What sort of metric space is developed in the categorical memories in the trained model network? It is highly desired to establish a direct relation between spatio-temporal analysis of neural activity associated with perception of categorical patterns, and intended action-ADO pairs. Although the present preliminary study used principle component analysis (PCA) [24] to examine how neural activations generated by the MSTRNN retain relationships between actions and corresponding ADOs, solid results have not yet been obtained. Future study should develop effective methods to determine a distance measure among spatio-temporal patterns in massive numbers of neural units for different categories.

6 Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2014R1A2A2A01005491).

References

[1] LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proceedings of the IEEE. 1998 Nov;86(11):2278-324.

[2] Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L. Imagenet: A large-scale hierarchical image database. The IEEE Conference on Computer Vision and Pattern Recognition; 2009 20 Jun. p. 248-255.

[3] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2015. p. 1-9.

[4] Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC, Fei-Fei L. Imagenet large scale visual recognition challenge. International Journal of Computer Vision. 2015 Dec 1;115(3):211-52.

[5] Ji S, Xu W, Yang M, Yu K. 3D convolutional neural networks for human action recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on. 2013 Jan;35(1):221-31.

[6] Soomro K, Zamir AR, Shah M. UCF101: A Dataset of 101 Human Action Classes From Videos in The Wild. arXiv preprint arXiv:1212.0402. 2012 Dec 3.

[7] Kuehne H, Jhuang H, Garrote E, Poggio T, Serre T. HMDB: a large video database for human motion recognition. In Proceedings of the International Conference on Computer Vision (ICCV). 2011.

[8] Simonyan K, Zisserman A. Two-stream convolutional networks for action recognition in videos. In Advances in Neural Information Processing Systems 2014 (pp. 568-576).

[9] Baccouche M, Mamalet F, Wolf C, Garcia C, Baskurt A. Sequential Deep Learning for Human Action Recognition. Human Behavior Understanding. 2011:29-39.

[10] Gers FA, Schraudolph NN, Schmidhuber J. Learning precise timing with LSTM recurrent networks. The Journal of Machine Learning Research. 2003;(3):115–43.

[11] Venugopalan S, Rohrbach M, Donahue J, Mooney R, Darrell T, Saenko K. Sequence to Sequence - Video to Text. CoRR. 2015;abs/1505.00487.

[12] Shi X, Chen Z, Wang H, Yeung DY, Wong WK, Woo WC. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In Advances in Neural Information Processing Systems 2015 (pp. 802-810).

[13] Arbib MA. Perceptual structures and distributed motor control: Bethesda, MD: American Physiological Society.; 1981. 1448-80.

[14] Tani J. Self-Organization and Compositionality in Cognitive Brains: A Neurorobotics Study. Proceedings of the IEEE. 2014;102(4):586-605. doi: 10.1109/JPROC.2014.2308604.

[15] Jung M, Hwang J, Tani J. Self-Organization of Spatio-Temporal Hierarchy via Learning of Dynamic Visual Image Patterns on Action Sequences. PloS one. 2015 Jul 6;10(7):e0131214.

[16] Felleman DJ, Van Essen DC. Distributed hierarchical processing in the primate cerebral cortex. Cerebral cortex. 1991 Jan 1;1(1):1-47.

[17] Hasson U, Yang E, Vallines I, Heeger DJ, Rubin N. A hierarchy of temporal receptive windows in human cortex. The Journal of Neuroscience. 2008 Mar 5;28(10):2539-50.

[18] Elman JL. Finding structure in time. Cognitive Science. 1990;14:179-211.

[19] Jordan MI. Attractor dynamics and parallelism in a connectionist sequential machine. Artificial neural networks: IEEE Press; 1990. p. 112-27.

[20] Yamashita Y, Tani J. Emergence of Functional Hierarchy in a Multiple Timescale Neural Network Model: A Humanoid Robot Experiment. PLoS Comput Biol 2008;4(11). doi: 10.1371/journal.pcbi.1000220.

[21] LeCun YA, Bottou L, Orr GB, Müller KR. Efficient backprop. In Neural networks: Tricks of the trade 2012 (pp. 9-48). Springer Berlin Heidelberg.

[22] Rumelhart DE, McClelland JL, Group PR. Parallel distributed processing: MIT press; 1986.

[23] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In advances in neural information processing systems 2012 (pp. 1097-1105).

[24] Hotelling H. Analysis of a Complex of Statistical Variables into Principal Components. Journal of Educational Psychology. 1933;24:417–41, 98–520.

[25] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research. 2014 Jan 1;15(1):1929-58.

[26] Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L. Large-scale video classification with convolutional neural networks. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition 2014. p. 1725-1732.

[27] Gal Y. A theoretically grounded application of dropout in recurrent neural networks. arXiv preprint arXiv:1512.05287. 2015 Dec 16.

[28] Cooijmans T, Ballas N, Laurent C, Courville A. Recurrent Batch Normalization. arXiv preprint arXiv:1603.09025. 2016 Mar 30.

[29] Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078. 2014 Jun 3.

[30] Graves A, Mohamed AR, Hinton G. Speech recognition with deep recurrent neural networks. In 2013 IEEE international conference on acoustics, speech and signal processing 2013 May 26 (pp. 6645-6649).

[31] Chung J, Gulcehre C, Cho K, Bengio Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555. 2014 Dec 11.

[32] Paine R.W. and Tani J. How hierarchical control self-organizes in artificial adaptive systems. Adaptive Behavior, Vol.13, No.3, pp.211-225, 2005.

[33] Goel A, Buonomano DV. Timing as an intrinsic property of neural networks: evidence from in vivo and in vitro experiments. Phil. Trans. R. Soc. B. 2014 Mar 5;369(1637):20120460.

[34] Bub D, Masson M. Gestural knowledge evoked by objects as part of conceptual representations. Aphasiology. 2006 Sep 1;20(9):1112-24.