Accepted in IEEE Transactions on Cognitive and Developmental Systems

# Recognition of Visually Perceived Compositional Human Actions by Multiple Spatio-Temporal Scales Recurrent Neural Networks

Haanvid Lee, Minju Jung, and Jun Tani

***Abstract*—We investigate a deep learning model for action recognition that simultaneously extracts spatio-temporal information from a raw RGB input data. The proposed multiple spatio-temporal scales recurrent neural network (MSTRNN) model is derived by combining multiple timescale recurrent dynamics with a conventional convolutional neural network model. The architecture of the proposed model imposes both spatial and temporal constraints simultaneously on its neural activities. The constraints vary, with multiple scales in different layers. As suggested by the principle of upward and downward causation, it is assumed that the network can develop a functional hierarchy using its constraints during training. To evaluate and observe the characteristics of the proposed model, we use three human action datasets consisting of different primitive actions and different compositionality levels. The performance capabilities of the MSTRNN model on these datasets are compared with those of other representative deep learning models used in the field. The results show that the MSTRNN outperforms baseline models while using fewer parameters. The characteristics of the proposed model are observed by analyzing its internal representation properties. The analysis clarifies how the spatio-temporal constraints of the MSTRNN model aid in how it extracts critical spatio-temporal information relevant to its given tasks.**

***Index Terms*—Action recognition, dynamic vision processing, convolutional neural network, recurrent neural network, symbol grounding.**

## I. Introduction

CONVOLUTIONAL neural networks (CNNs) [1], inspired by the mammalian visual cortex, show remarkably better object image recognition performance than conventional vision recognition schemes which employ elaborately hand-coded visual features. A CNN trained with one million visual images from ImageNet [2] was able to classify hundreds of object images with an error rate of 6.67% [3], demonstrating near-human performance [4]. However, CNNs lack the capacity for temporal information processing. Thus, CNNs are less effective when used to handle video image patterns as compared to static images.

To address this shortcoming, a number of action recognition models have been developed. Typical deep learning models for action recognition are 3D convolutional neural networks (3D-CNNs) [5], long-term recurrent convolutional networks (LRCNs) [6], and two-stream convolutional networks [7]. The 3D-CNN extracts the spatio-temporal features of videos through convolutions in the temporal and spatial domains in a fixed window [5]. The LRCN is a two-stage model that initially extracts spatial features in its CNN stage and then extracts temporal features from its long-short term memory (LSTM) [8] stage [6]. And the two-stream convolutional network has one CNN stream for RGB input and another CNN stream for the input of stacked optical flows. The two streams are joined at the end to create a categorical output [7].

Although 3D-CNNs, LRCNs, and two-stream convolutional networks perform well, some of their dynamics are not backed by neuroscientific findings. One important piece of evidence in mammals is that the size of the spatio-temporal receptive field of each cell is increased as the level goes higher [9], [10]. Moreover, the principle of downward causation [11], [12], which comes from the cybernetics era, suggests that a spatio-temporal hierarchy can be naturally developed in the human brain by taking advantage of the macroscopic constraints genetically assigned to them. This evidence and principle suggest that a deep learning model for action recognition should form a hierarchy by the assignment of spatio-temporal constraints. The model should also extract spatial and temporal features simultaneously considering the hierarchy. However, typical action recognition models lack these capabilities.

The current study is an extension of the multiple spatio-temporal scales neural network (MSTNN) [13]. The neural activities of the MSTNN are governed by spatial and temporal constraints which correspondingly work via the local connectivity of convolutional layers and time constants assigned to leaky integrator neural units at each layer [13]. These constraints allow the MSTNN to develop faster dynamics and local interactions in the lower layer, whereas it develops slower and global interactions at the higher level. This enables the MSTNN to extract spatio-temporal features in multiple spatio-temporal abstractions from its input videos. This formation of a spatio-temporal hierarchy is consistent with the biological evidence and the principle mentioned earlier.

However, the temporal processing capacity of the MSTNN is quite limited given the fact that its essential dynamics is the decay dynamics exerted by the leaky integrator neurons [13] that compose the model. At the same time, the MSTNN only uses forward connectivity without any recurrent structures, while there is biological evidence suggesting that the primary

H. Lee is with the School of Computing, Korea Advanced Institute of Science and Technology (KAIST), Daejeon 305-701, Republic of Korea, e-mail: (haanvidlee@gmail.com).

M. Jung is with the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon 305-701, Republic of Korea, e-mail: (minju5436@gmail.com).

J. Tani is with the Cognitive Neurorobotics Research Unit, Okinawa Institute of Science and Technology (OIST), Okinawa 904-0412, Japan, e-mail: (tani1216jp@gmail.com). The correspondence should be sent to J. Tani.

visual cortex of a cat has a recurrent connectivity that is used for identifying past and current visual inputs [14], [15]. In this context, the current study attempts to add recursive dynamics to the MSTNN by introducing recurrent connectivity in its convolutional layers. This leads to our novel proposal of the MSTRNN model in the current study.

In the experiments, MSTRNNs were compared with MSTNNs and LRCNs. MSTRNNs were compared to the network without recurrent connections (MSTNNs) to observe how the existence of the recurrent structure in the models affects the action recognition performances. Among the introduced representative action recognition models, networks with sequentially connected stages of CNNs and LSTMs (LRCNs) were also used as baselines. The LRCN was chosen as the baseline model of the MSTRNN to determine how spatio-temporal constraints and the simultaneous extraction of spatial and temporal features of the MSTRNN makes it different from the LRCN.

The MSTRNN model was compared to the baselines (MSTNN and LRCN) and evaluated using three different human action datasets that are distinct in terms of the types and levels of compositionality introduced in the action patterns. ”Compositionality” refers to the degree of possible composition/decomposition of one whole pattern by/into reusable parts. For a machine vision system to recognize human actions with semantics, it must perceive videos of actions while extracting the underlying spatio-temporal compositionalities of the actions. Such spatio-temporal structures should be linked to compositionality during the generation of human actions [16], [17]. Temporal compositionality can be accounted by the fact that most goal-directed human actions are composed of sequential combinations of commonly used behavior primitives [16]. Spatial compositionality can be accounted by combinations of transitive actions and objects in object-directed actions or coordinated combinations of movement patterns of different limbs of a person. The challenge with regard to a visual understanding of human action is to extract such compositional structures under the condition that each trajectory of the perceived visual stream can be diverse, even for an identical category of action, as the profiles of behavior primitives are quite deformational depending on the individual. The three actions concatenated Weizmann dataset (3ACWD), as used in our first experiment, is created by concatenating three actions from the Weizmann dataset [18] in a sequence. The second and the third datasets created for the purpose of this study have natural human action patterns with the levels of the underlying compositionalities made to be higher than in the first dataset. The second dataset, which is the compositionality level 1 action dataset (CL1AD), contains actions with objects and has action-directed-object (ADO) and action categories. The third one, which is the compositionality level 2 action dataset (CL2AD), has an increased level of compositionality relative to that of the second dataset given its ADO, action, and modifier categories. Here, a modifier refers to words that modify actions. We test the MSTRNN with the baselines (MSTNN and LSTM) on these datasets of different compositionality levels to test their performance outcomes and to determine how their categorical memories are formed after training.

## II. PROPOSED MODEL

The MSTRNN model consists of the following six types of layers: input, convolutional and pooling layers, context layers, fully-connected layers, and an output layer as shown in Fig. 1 (A). The MSTRNN receives a stream of RGB images in the input layer. If the input image frame is large, it can be reduced by going through several convolutional and pooling layers in a sequential manner. The context layers then extract the spatio-temporal features. The extracted spatio-temporal features pass through several fully-connected layers. Finally, the categorical output is realized by a fully-connected layer with softmax activation.

The context layer of a MSTRNN 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 in Fig. 1 (B). If the block that represents context units is deleted from Fig. 1 (B), Fig. 1 would be an illustration of a MSTNN structure. The context units, which have recurrent connectivity, are the only difference between the MSTRNN and the MSTNN. 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 causes the internal states of the leaky integrator neurons in the context layer to change more slowly at each time step [19]. The MSTRNN assigns a larger time constant to higher layers to develop a spatio-temporal hierarchy [13], [19].

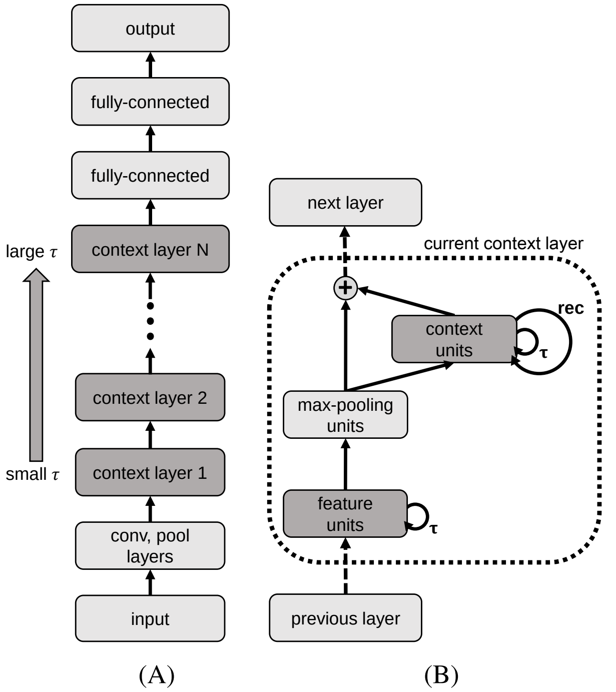


Fig. 1. The architecture of the MSTRNN. (A) The full architecture of the MSTRNN. $\tau$ is a time constant for a set of leaky integrator neurons in a context layer. (B) The structure of the context layer. The layers/units with memories are more shaded. The arrow labeled $\tau$ indicates the decay dynamics of the leaky integrator neurons in the feature units and context units. The word *rec* with the arrow indicates recurrent connections made in the context units.

### A. Batch Normalization

When training deep neural networks, internal covariate shift problem arises. The training of a layer depends on the output of the previous layer, but the distribution of neuronal activations generated by each layer changes after each update of the weights. This causes the training of deep neural networks to be slow and difficult.

To alleviate the internal covariate shift problem, Ioffe et al. presented the batch normalization (BN) method [20]. It was reported in their work that BN accelerates training of deep neural networks. They also showed that BN enhances the performance of neural networks. The method normalizes each neuronal activation to zero mean and unit variance, as described below,

$$\mu \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i, \tag{1}$$

$$\sigma \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i-\mu)^2, \tag{2}$$

$$\hat{x}_i \leftarrow \frac{(x_i-\mu)^2}{\sqrt{\sigma^2+\epsilon}}, \tag{3}$$

$$y_i \leftarrow \gamma\hat{x}_i+\beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i), \tag{4}$$

where $m$ is the size of a mini-batch, and the activation of a neuron is represented by $x$. BN first computes the mean ($\mu$) and standard deviation ($\sigma$) of the mini-batch activations via (1) and (2), respectively. It then normalizes the activation as described in (3). In the equation, the small positive constant $\epsilon$ is added to the variance to maintain numerical stability. The normalized value $\hat{x}_i$ is then scaled and shifted by trainable parameters $\gamma$ and $\beta$ as in (4). The process of scaling and shifting enables the BN method to restore the original activations if necessary.

After BN was demonstrated to be effective in dealing with the internal covariate shift problem in feed forward networks, Cooijmans et al. showed that BN can also be applied to recurrent neural networks [21]. In their work, they applied BN not only to the input that is fed forward, but also to the input that is fed in a recurrent manner.

In our work, we used BN on the feed forward layers before the activation functions. We also used the recurrent BN method on the feature units and context units of the context layers to help the proposed model generalize better and to accelerate the training. For the LRCNs used in this experiment, BN and recurrent BN were applied in the manner Ioffe et al. and Cooijmans et al. suggested in their works. How the BN and recurrent BN approaches are applied to the MSTNN is explained in the next subsection (*B. Feature Units*).

### B. Feature Units

The feature units are capable of extracting temporal features via the decay dynamics of the leaky integrator neurons. They are also able to extract spatial features by convolutional operations [13]. The forward dynamics of the feature units are explained in (5) and (6). The internal state and the activation value of a neuron at the *l*th context layer, the *m*th map of feature units, the retinotopic coordinates (x, y), and at time step *t* are represented as $\hat{f}_{lm}^{txy}$ and $f_{lm}^{txy}$ respectively.

$$\begin{aligned}\hat{f}_{lm}^{txy} &= \Big(1-\frac{1}{\tau_l}\Big)\hat{f}_{lm}^{(t-1)xy} \\ &\quad+\frac{1}{\tau_l}\Big(\mathrm{BN}_{\gamma,\beta}\Big(\sum_{n=1}^{N_{l-1}}(k_{lmn} * p_{(l-1)n}^{t})^{xy}\Big)+b_{lm}\Big) \\ &\quad+\frac{1}{\tau_l}\mathrm{BN}_{\hat{\gamma},\hat{\beta}}\Big(\sum_{a=1}^{A_{l-1}}(z_{lma} * c_{(l-1)a}^{t})^{xy}\Big),\end{aligned} \tag{5}$$

$$f_{lm}^{txy} = \max(0, \hat{f}_{lm}^{txy}). \tag{6}$$

Here, $\tau$ represents the time constant and $k$ and $z$ are the convolutional kernels that extract the features from the pooling units and context units, respectively, of the previous layer. Additionally, *b* represents the bias used in the convolution operation, $*$ is the convolution operator, *N* is the total number of maps of the pooling units, and *A* is the total number of maps of the context units. Additionally, *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 (5) 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. BN is applied to the feed forward paths (the second and third terms of (5)). Equation (5) also describes the dynamics of the MSTNN when the third term is discarded. Equation (6) shows the rectified linear unit (ReLU) that serves as the activation functions of the feature units. ReLU is reported to accelerate the training of a convolutional layer given that it is a non-saturating nonlinear function [22].

### C. Context Units

A set of context units (Fig. 1 (B)) is equivalent to feature units with recurrent convolutional kernels. Due to the addition of recurrent convolutional kernels in the structure of the feature units, the temporal processing capacity of the context units is enhanced compared to that of the feature units. The recurrent dynamics of the context units enhances the extraction of latent temporal features from input image sequences [23], [24]. The recurrent connections are made by convolutional kernels. The forward dynamics of the context units are shown in (7). The internal state of a neuron in the (x, y) coordinates of the *a*th map of the context units in the *l*th context layer, at time step *t* are represented as $\hat{c}_{la}^{txy}$.

$$\begin{aligned}\hat{c}_{la}^{txy} &= \Big(1-\frac{1}{\tau_l}\Big)\hat{c}_{la}^{(t-1)xy} \\ &\quad+\frac{1}{\tau_l}\mathrm{BN}_{\gamma,\beta}\Big(\sum_{m=1}^{M_l}(\tilde{k}_{lam} * p_{lm}^{t})^{xy}+\tilde{b}_{la}\Big) \\ &\quad+\frac{1}{\tau_l}\mathrm{BN}_{\hat{\gamma},\hat{\beta}}\Big(\sum_{b=1}^{B_l}(\tilde{z}_{lab} * c_{lb}^{t-1})^{xy}\Big).\end{aligned} \tag{7}$$

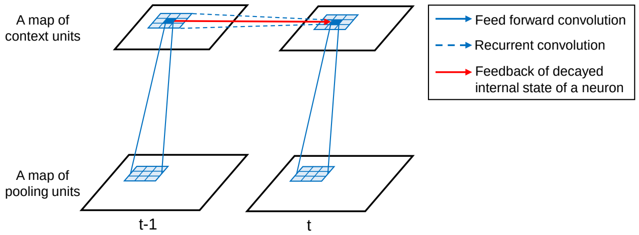


Fig. 2. Illustration of the dynamics in context units at the time steps of t-1 and t.

Here, $c$ denotes the activation value of a context unit, $\tilde{k}$ is the convolutional kernel, $\tilde{b}$ is the bias for the convolution operation, $M$ is the total number of maps of the pooling units, $B$ is the total number of maps of the context units, $\tilde{z}$ is the recurrent convolutional kernel of the context units, and $p$ is the neural activations of the pooling units. The first term on the right hand side of (7) describes the decay dynamics of the leaky integrator neurons. The dynamics is illustrated in Fig. 2, indicated by the red arrow. The second term represents the convolution of the pooling units. The feed forward convolution is represented by the solid lines in Fig. 2. The third term describes the recurrent dynamics in terms of the recurrent shared weights. The recurrent convolution operation is represented by the dotted lines in Fig. 2. The neural activations of the context units in the previous time step are supplied through the recurrent convolutional kernels. BN is applied to the feed forward path (the second term) and is also applied to the recurrent path (the third term). ReLU is used as the activation function of the context units.

### D. *Fully-Connected Layers*

The hyperbolic tangent function recommended by LeCun et al. [25] is used as the activation function of the fully-connected layers,

$$a = 1.7159\tanh(\frac{2}{3}h), \tag{8}$$

where $h$ is the internal value of a neuron and $a$ denotes the activation of the neuron. The function gives outputs of 1 and -1 for inputs of 1 and -1, respectively. At 1 and -1, the function has maximum absolute values for its second derivatives. It was reported in the work by LeCun et al. that this characteristic helps a deep learning model to converge its performance near the end of its training [25].

### E. *Training Method*

All the videos in a dataset are padded with its last image frame until the frame length of each video is equal to the maximum video frame length found in the dataset. The MSTRNN is trained on the errors that are generated in the last 15 time steps of the videos. This is because we used video datasets which action categories are identifiable only at the end. Hereafter, the last 15 steps are referred to as the voting period. The cost function is the sum of the negative log likelihood errors that were calculated for each time step during the voting period,

$$E = -\frac{1}{L}\sum_{t=T-L+1}^{T}\sum_{n=1}^{N} y_{n,t}\log\hat{y}_{n,t}, \tag{9}$$

where $E$ is the error obtained from an action video, $L$ is the voting period, $T$ is the video duration, $N$ is the number of categories, $\hat{y}_n$ is the model output, and $y_n$ is the true label. Because the networks examined in our study have memories, the errors are back propagated through time [26] to optimize the learnable parameters. The MSTRNN is also trained by means of mini-batch stochastic gradient descent. The mini-batch sizes are 27, 90, and 84 for the first, second, and third experiment described in the section *III. EXPERIMENTS*. The Adam optimizer [27] was also used for parameter optimization.

To prevent overfitting, all learnable parameters (except biases) were learned with a weight decay of 0.0005 [22]. In addition, random cropping [28] of the input images, making them ten pixels smaller by width and height, was also used to avoid overfitting. To alleviate the internal covariate shift problem, the BN [20] and recurrent BN [21] methods were used. For $\gamma$ and $\beta$ used in the methods, their initial values were set to 0.1 and 0, respectively, suggested by Cooijmans et al. [21]. Gradients are clipped in the range [-12, 12] to provide stability during the training [29]. For the training of the LRCNs, the forget gate values were initially set to 1 to encourage remembering and to speed up the training [30].

The learning rate was 0.01 for the first 200 epochs and was then lowered to 0.005 and 0.001 for the next 100 epochs and the last 300 epochs, respectively, in the experiment with 3ACWD. In the second and third experiments, the learning rate was 0.005 for the first 400 epochs and was then lowered to 0.001. The MSTRNNs and baselines were trained until their recognition accuracies converged.

### F. *Performance Evaluation*

To evaluate the categorization performance, the leave-one-subject-out cross-validation scheme was used. In this method, one subject was selected from the dataset and his/her video clips were left out of the training data, to be used as the test data. The test videos were padded with its last image frame until their frame lengths reached the maximum frame length found in the dataset. Recognition accuracies obtained from all possible validation sets were averaged to be used as an evaluation measure. When there was more than one set of categories, such as actions and action-related objects, the recognition accuracy for the action-ADO pair was computed. The epoch for which the model showed the best accuracy in the joint category was then selected. The accuracy of each set of categories (i.e., objects and actions for the second experiment described in the section *III. EXPERIMENTS*) at the epoch was used for evaluation. The accuracies were rounded off to the second decimal place and recorded as measures of the overall performance of an action recognition model.

## III. Experiments

In all experiments, the MSTRNN is compared with the baseline models (MSTNN, LRCN). The first experiment was conducted using a relatively simple action dataset. Then, in the second and third experiments, the MSTRNN and the baseline models were tested on datasets that look more natural and have higher compositionality levels.

### A. Model Parameter Settings

The MSTRNN, MSTNN, and LRCN have identical input and output layers according to the datasets with which they are tested. The datasets are described in detail in the *Dataset* section inside sections *B*, *C*, and *D* of section *III. EXPERIMENTS*. Table I shows the RGB input image sizes according to the datasets after cropping, as described in the *Training Method* section.

A convolutional layer and a pooling layer were used as a preprocessing stage to decrease the computing time of the models by decreasing their input size, as shown in Fig. 3. The first convolutional kernel of the MSTRNN, MSTNN, and LRCN had a kernel size of 3x3 and a stride of 2x2 for 3ACWD and a stride of 3x3 for the other two datasets to decrease the input feature size. With few exceptions, most of the convolutional kernels used in the models have a size of 3x3 and a stride of 1x1. And all pooling kernels used in the models have a size of 2x2 and a stride of 2x2. The convolutional kernel was 3x3 and the pooling kernel was 2x2 because these sizes were found to be best for CNNs [31].

TABLE I
Input and Output Of The Tested Models

| Dataset | Input Size | Output Category | Softmax Neurons |
|---|---|---|---|
| 3ACWD | 73x73 | Action | 27 |
| CL1AD | 108x108 | ADO | 4 |
| | | Action | 9 |
| CL2AD | 108x108 | ADO | 4 |
| | | Action | 4 |
| | | Modifier | 6 |

**MSTRNN**: The MSTRNN model has one convolutional layer, one pooling layer, two context layers, two fully-connected layers, and a softmax layer, as described in Fig. 3 (A). The first and second context layers have time constants of 2 and 100, respectively, as Jung et al. [13] assigned to the first and last convolutional layers of the MSTNN used in their work. Sensitivity analysis of the time constants of the MSTRNN was conducted, and the result is discussed in the section *V. APPENDIX*. The result supports the assignment of the time constants.

Convolutional kernels that connect to context units (see Fig. 1 (B)) recurrently have a kernel size of 3x3 with a stride of 1x1 along with 1x1 zero padding. These convolutional kernels were used so that the resulting map sizes of the context units remain unchanged as the time steps progress. The convolutional kernels that connect the pooling units to the context units and the context units to the next layer have a size of 2x2 with a stride of 1x1.

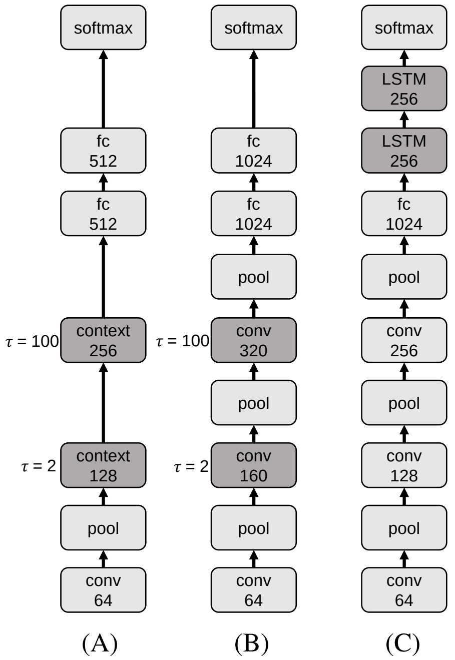


Fig. 3. Architectures of the models used in the experiments. (A) Architecture of the MSTRNN. (B) Architecture of the MSTNN. (C) Architecture of the LRCN. The layers that have memory are shaded.

**Baselines**: The MSTNN model has three convolutional layers, three pooling layers, two fully-connected layers, and a softmax layer (Fig. 3 (B)). The second and third convolutional layers consist of leaky integrator neurons that have time constants of 2 and 100, respectively.

The LRCN has three convolutional layers, three pooling layers, one fully-connected layer, two LSTM layers, and a softmax layer (Fig. 3 (C)). The original authors of the paper introducing the LRCN model reported that LRCNs having one fully-connected layer before the LSTM stage showed better performance than having two fully-connected layers [6]. Therefore, we used one fully-connected layer in the LRCN as shown in Fig. 3 (C). Although Donahue et al. [6] used the LRCN with one LSTM layer for an action recognition task, here we assigned two LSTM layers in the LRCN for a fair comparison with the MSTRNN and MSTNN, both of which have two temporal information processing layers (i.e., context layers of the MSTRNN and convolutional layers with leaky integrator neurons of the MSTNN). We also used 256 LSTM hidden units for the two LSTM layers because Donahue et al. reported that increasing the LSTM hidden units beyond 256 did not bring about a performance boost when the model was given RGB images as input.

The parameters of the MSTNN and LRCN are compared with the MSTRNN in Table II. The numbers were rounded up to the nearest ten thousand. The numbers of parameters in the three models differ subtly from those in experiments due to the different numbers of outputs depending on a dataset (Table I).

TABLE II
NUMBER OF PARAMETERS IN THE TESTED MODELS

| MSTRNN | LRCN | MSTNN |
|---|---|---|
| 3M | 4.6M | 4.6M |

### B. Categorization of the Three Actions Concatenated Patterns from the Weizmann Dataset

This section uses 3ACWD, which is less compositionally complex than CL1AD and CL2AD, to compare the characteristics of the proposed model (MSTRNN), the model without recurrent weights (MSTNN), and the model with separate spatial and temporal information processing stages with no temporal constraints (LRCN).

**Dataset**: A set of compositional action videos was prepared by concatenating videos of three different human actions from the Weizmann dataset [18]. The three actions were jump-in-place (JP), one-hand-wave (OH), and two-hand-wave (TH), as shown in Fig. 4, resulting in 27 categories, and one video clip of the concatenated actions for each category. 27 videos for each of the nine subjects exist in the dataset. The videos of the original dataset had frame rates of 25 and frame sizes of 144x180. After the concatenation of the videos, we resized the concatenated video frames to 83x83 and downsampled their frame rates by half of the original rates.

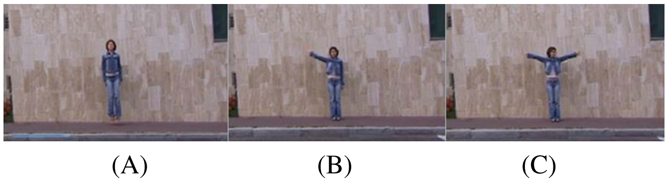


Fig. 4. The three human action categories used from the Weizmann dataset: (A) Jump-in-place. (B) One-hand-wave. (C) Two-hand-wave.

**Results**: The action categorization accuracy of the MSTRNN on 3ACWD was highest by the mean recognition rate of 89.30% with the standard error of 4.60% (Table III). This result demonstrates that context units with recurrent weights improve the categorization of long concatenated human action sequences. It also implies that the simultaneous extraction of spatio-temporal features with constraints (MSTRNN) is more beneficial during video processing as compared to separate extraction of spatial and temporal features with no temporal constraints (LRCN).

TABLE III
ACTION RECOGNITION ACCURACIES ON 3ACWD IN PERCENTAGES

| | MSTRNN | LRCN | MSTNN |
|---|---|---|---|
| Accuracy | **89.30±4.60** | 86.42±5.52 | 48.15±5.69 |
| Chance Level | 3.70 | | |

Next, the internal dynamics of MSTRNN, MSTNN, and LRCN were assessed by a time series analysis of their neural activation values. A test subject's videos were given as input to the models and their activation values were obtained from the second LSTM layer of the LRCN, the second convolutional layer among the two convolutional layers that are composed of leaky integrator neurons in the MSTNN, and the context units in the second context layer of the MSTRNN (see Fig. 3). The time series neural activations were visualized by means of a principle component analysis (PCA) [32]. First three principle components of the neural activations were used for the visualization. In the following discussion of this analysis, the PCA mapping of the time series neural activations obtained when given action primitives A, B, and C in a sequential manner is designated as PCA trajectory A-B-C. Because all models were trained to categorize actions based on the outputs obtained during the voting period, the last 15 positions of the trajectories should be differentiated based on the history of the images that were shown in sequence.

We compare the PCA trajectories of the MSTRNN, MSTNN, and LRCN obtained from the three input videos that have identical actions for all three primitive actions (JP-JP-JP, OH-OH-OH, TH-TH-TH), and two videos that have OH and TH as their first action primitives and have JP for their second and third action primitives (OH-JP-JP, TH-JP-JP). The PCA trajectories obtained from the three models are shown in Fig. 5. The first, second, third columns of Fig. 5 shows the trajectories drawn with the first and second principle components, the first and third principle components, and the second and third principle components respectively.

For all models (MSTRNN, MSTNN, LRCN), the PCA trajectories of OH-OH-OH and OH-JP-JP are very similar to one another while the images of the first primitive action (OH) are fed into the models, as shown in Fig. 5. In addition, the PCA trajectories of TH-TH-TH and TH-JP-JP have similar characteristics in terms of the trajectories of OH-OH-OH and OH-JP-JP. But the trajectories of OH-JP-JP and TH-JP-JP take different paths from those of OH-OH-OH and TH-TH-TH, respectively, when images of the second action primitives are used as input to the models.

For the MSTNN, the trajectories of OH-JP-JP and TH-JP-JP approach the end of the JP-JP-JP trajectory, as shown in Fig. 5 (B). The decay dynamics of the MSTNN is responsible for its development of similar activation values for the input videos of JP-JP-JP, OH-JP-JP, and TH-JP-JP, while JP was input to the model for second and third primitive actions. The internal neural values of the convolutional layers composed of leaky integrator neurons in the MSTNN are affected by both the current feed forward input and the decayed internal neural values of the previous time step. Because the spatio-temporal information of the past gradually decays over time, spatio-temporal features extracted in the past cannot effectively influence the internal neural values of the current time step to monitor which action primitives came in the past.

For the MSTRNN, the ends of the OH-JP-JP and TH-JP-JP trajectories did not converge to the end point of the JP-JP-JP trajectory. It is clearly shown in Fig. 5 (A), specifically with regard to the PCA trajectories drawn with the first and second principle components, that the end positions of OH-JP-JP and TH-JP-JP do not converge to the end point of JP-JP-JP. This shows that the MSTRNN has better categorical memory than the MSTNN because it was able to use the spatio-temporal information of the previously shown first action primitives of

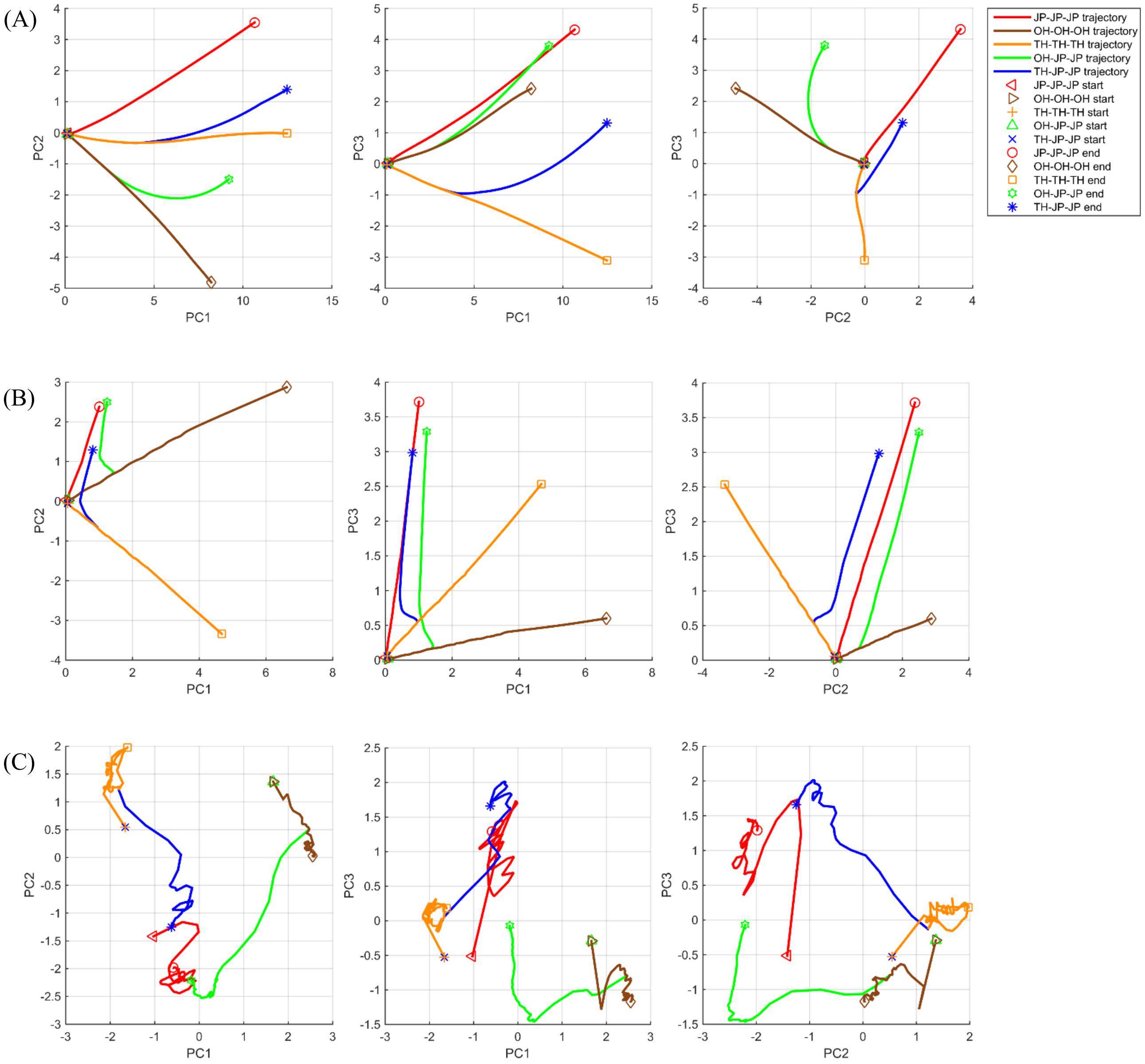


Fig. 5. PCA mapping of the time series activation values obtained from the MSTRNN, MSTNN, and LRCN when the test subject′s action videos of the JP-JP-JP, OH-OH-OH, TH-TH-TH, OH-JP-JP, and TH-JP-JP concatenated action sequences were given as input to the models. (A) PCA mapping generated from the context units in the second context layer of the MSTRNN. (B) PCA mapping generated from the second convolutional layer among the two convolutional layers that are composed of leaky integrator neurons in the MSTNN. (C) PCA mapping generated from the second LSTM layer in the LRCN.

the OH-JP-JP and TH-JP-JP trajectories to differentiate the end positions of the JP-JP-JP, OH-JP-JP, and TH-JP-JP trajectories.

The LRCN showed chaotic PCA trajectories that are similar to the trajectories stemming from the Brownian motion of particles (Fig. 5 (C)). Moreover, the end points of OH-JP-JP and TH-JP-JP approached the end point of JP-JP-JP, as in the trajectories obtained from the MSTNN. This suggests that although the LSTM is regarded to self-adjust the timescales of its own dynamics with its forgetting gates, performing well in generating/recognizing low dimensional sequential data [33], [34], [35], its performance cannot be guaranteed with high dimensional temporal data. This result implies that it is beneficial to simultaneously extract spatial and temporal features and introduce multiscale constraints on the operation.

### *C. Categorization of the Compositionality Level 1 Action Dataset*

This experiment tested the MSTRNN, MSTNN, and LRCN with the newly prepared CL1AD. CL1AD has a higher compositionality level than the datasets with only action categories (e.g., 3ACWD). CL1AD contains videos of subjects manipulating objects, and the dataset has categories for actions and ADOs.

**Dataset**: CL1AD consists of 900 videos made by ten subjects performing nine actions directed at four objects. There are 15 object-directed action categories in total (Fig. 6). The dataset was designed so that the categorization task was non-trivial. A non-ADO (distractor) appears along with an ADO in

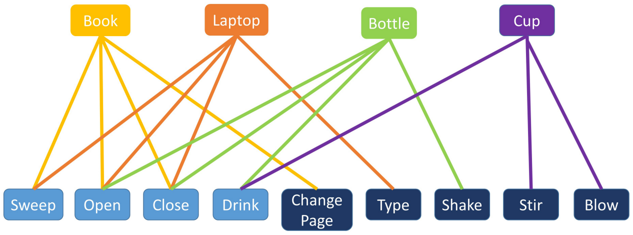


Fig. 6. Composition of CL1AD. 15 classes were made by combination of 4 objects and 9 actions.

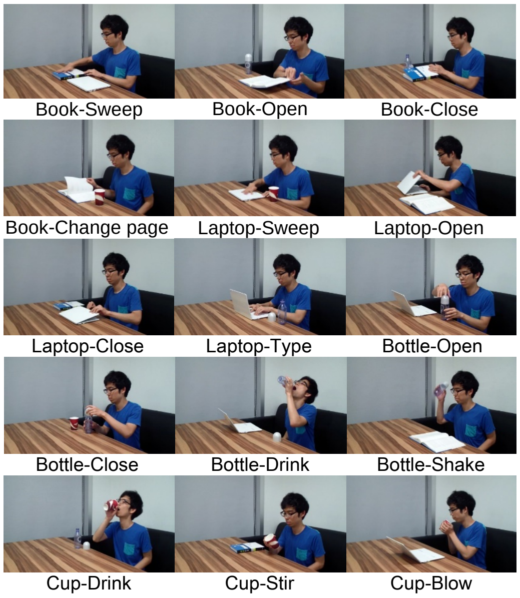

Book-Sweep
Book-Open
Book-Close
Book-Change page
Laptop-Sweep
Laptop-Open
Laptop-Close
Laptop-Type
Bottle-Open
Bottle-Close
Bottle-Drink
Bottle-Shake
Cup-Drink
Cup-Stir
Cup-Blow

Fig. 7. Sample frames from CL1AD. An object that is not related to the action also appears in the scene to make the recognition of ADO non-trivial.

each video to prevent the model from inferring a human action or an ADO solely by recognizing an object in a video (Fig. 7). For each object-directed action category, six videos were shot for each subject with three different non-ADOs appearing in two videos each in different states (opened or closed) if possible, as they are presented in the other videos as ADOs. During the recording of the dataset, the subjects generated each action without constraints. Objects were located at random positions in the task space. However, the camera view angle was fixed, as the problem of view invariance is beyond the scope of the current study. The original video frame rate is 60 and the frame size is 480x720. For our use in the experiment, we have downsampled the videos by a rate of 6 and resized the frames by 118x118. The dataset is open to the public (available at https://github.com/haanvid/CL1AD/releases).

**Results**: The MSTRNN again showed the highest categorization performance relative to the baselines (LRCN, MSTNN) on CL1AD, which has a higher compositionality level than 3ACWD (Table IV). This confirms that the MSTRNN, which is characterized by its multiscale spatio-temporal constraints, the simultaneous extraction of spatio-temporal features, and recurrent connectivity, outperforms the model without the recurrent connectivity (MSTNN) and the model that lacks the capacity to extract spatio-temporal features simultaneously and multiscale temporal constraints (LRCN).

TABLE IV
RECOGNITION ACCURACIES ON CL1AD IN PERCENTAGES

| | | MSTRNN | LRCN | MSTNN |
|---|---|---|---|---|
| ADO | Accuracy | **95.67±1.72** | 92.67±2.13 | 85.44±4.85 |
| | Chance Level | 25.00 | | |
| Action | Accuracy | **93.56±3.00** | 91.11±2.42 | 76.78±5.31 |
| | Chance Level | 11.11 | | |

Next, we observe how the characteristics of the MSTRNN helped with the action recognition task by comparing the time series activations of the MSTRNN to that of the baselines (MSTNN and LRCN). Given a test subject's data on the Sweep/Close-Laptop ADO-action joint category, the neural activations of the models were visualized by PCA in a manner similar to that in the previous experiment. The PCA trajectories of the Sweep/Close-Laptop videos are visualized and reported because the images and movements in the videos differ in their early stages but become similar as the videos play near their ends. In the early stages of the videos, different actions of sweeping and closing the laptop are displayed in the image streams. But near the ends of the videos for both categories, similar actions of a test subject touching a closed laptop and then moving their hands away from the laptop to their knees are shown. Therefore, the videos from the classes of Sweep/Close-Laptop require an action recognition model to classify them using the extracted spatio-temporal features from the early stages of the videos

The PCA trajectories obtained from the MSTRNN (Fig. 8 (A)) show that the model is capable of remembering the spatio-temporal features of the closing and sweeping motion in the early stage of the videos. The ends of the Sweep-Laptop PCA trajectories and Close-Laptop trajectories are clustered distinctively and according to their categories, especially in the PCA plot drawn with the first and second principle components.

The ends of the PCA trajectories of the LRCN (Fig. 8 (C)) also appear to cluster according to the class. However, their trajectories appear to make oscillating movements, sometimes even going back and forth. These phenomena may have occurred because the LSTM could not learn to develop a spatio-temporal hierarchy due to separate extraction of spatial and temporal features and the lack of constraints imposed during the process. This may be why the categorization performance of the LRCN is lower than that of the MSTRNN.

For the MSTNN, the trajectories are mixed and are not differentiated well (Fig. 8 (B)). This occurs due to the lack of the recurrent pathways that exist in the other models (MSTRNN, LRCN). The trajectories generated by the MSTNN explain the lower performance of the MSTNN on the recognition task compared to the other models.

### D. Categorization of the Compositionality Level 2 Action Dataset

In the third experiment, the categorization performances of the MSTRNN and the baseline models (MSTNN and LRCN)

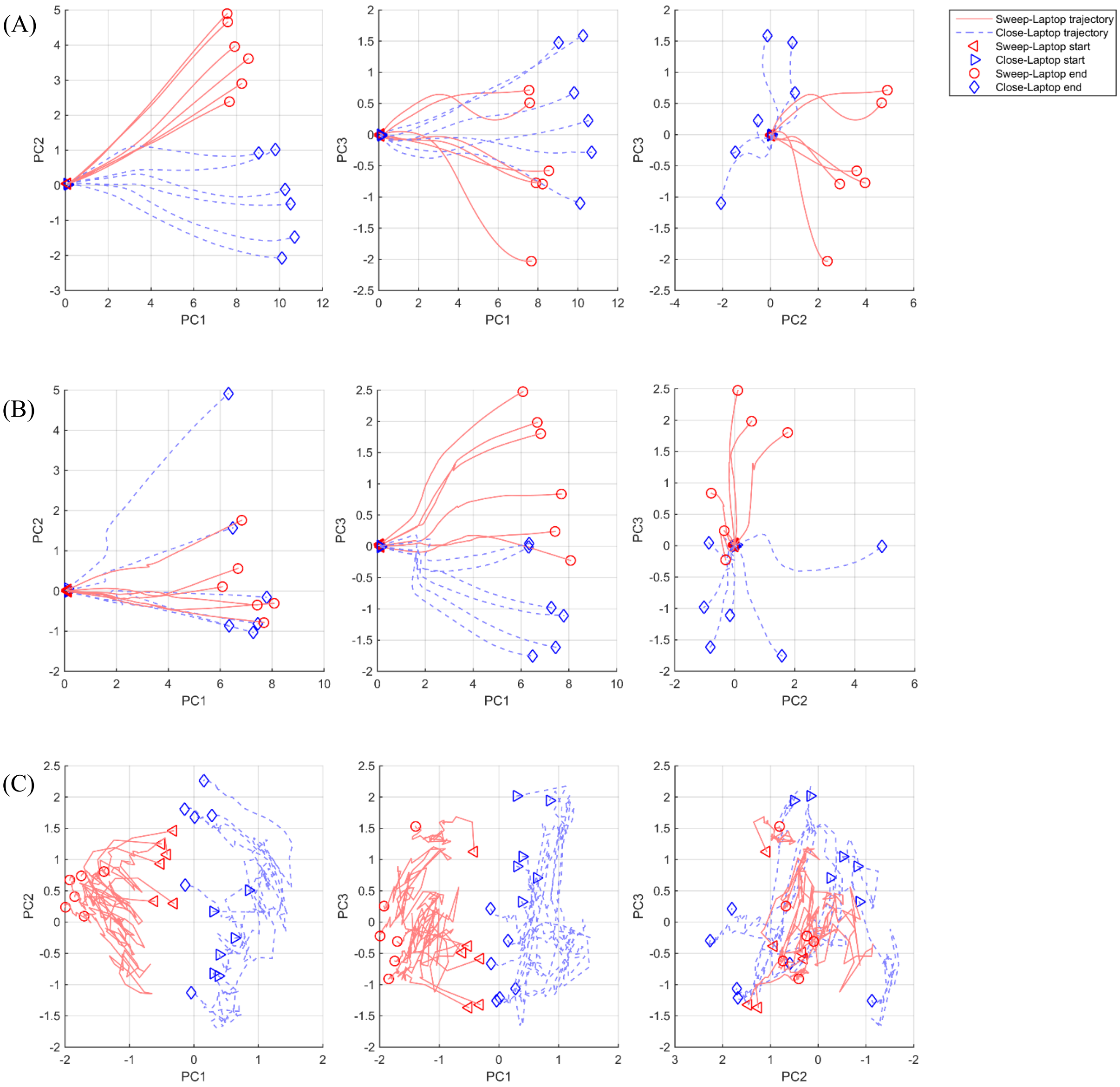


Fig. 8. PCA trajectories obtained from Sweep/Close-Laptop. (A) Trajectories of the MSTRNN. (B) Trajectories of the MSTNN. (C) Trajectories of the LRCN.

were compared by introducing a more challenging task using CL2AD. Each action pattern in CL2AD can be expressed by an object, an action, and an action modifier.

**Dataset**: CL2AD consists of 840 videos that are describable according to the composition of four objects, four actions, and six action modifiers. The assumed categories in the dataset in terms of the object-action-modifier triplets are shown in Fig. 9. The total number of triplets is 42. The video recordings were taken from ten different subjects. A subject shot two videos for each object, action, and modifier triplet. During the dataset recording process, the subjects generated each action naturally. As in CL1AD, one ADO and one distractor appear in each video of CL2AD. And distractors were randomly chosen. Although objects were located in various positions in the task space, the camera view angle was fixed (the problem of view invariance is beyond the scope of the current study). The frame rate and frame size are identical to those with CL1AD. CL1AD is also downsampled by a rate of 6, and the frame size was resized to 118x118. The dataset is open to the public (available at https://github.com/haanvid/CL2AD/releases).

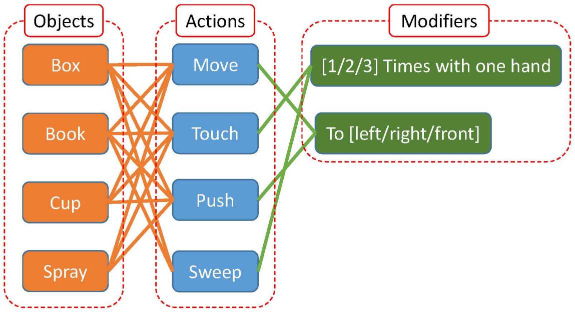
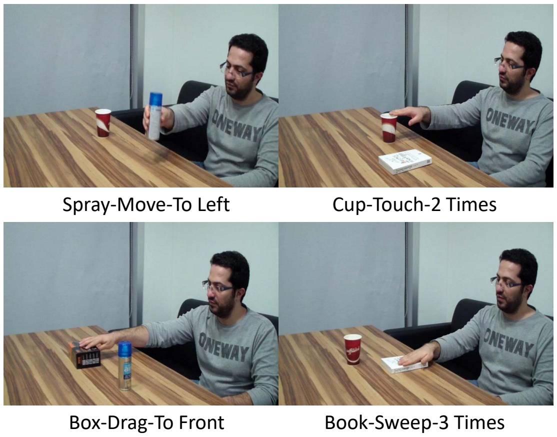


Fig. 9. Composition of the CL2AD. Each object-action-modifier triplet or joint category is indicated by a connection. There are 42 joint categories.



Spray-Move-To Left

Cup-Touch-2 Times

Box-Drag-To Front

Book-Sweep-3 Times

Fig. 10. Sample frames from CL2AD. It is difficult to infer the action category from a static image. However, it is even more difficult to infer the action modifier from the image. Recognition of action modifiers requires an action recognition model to extract a longer temporal correlation as compared to that from the recognition of actions or ADOs.

The categorization of action modifiers often requires the action recognition models to extract longer temporal correlations than in the case when only an action category is required. For example, for a video in the Cup-Touch-2 Times joint category, because the model can acquire sufficient information only after perceiving a cup twice touched, the model must extract a long-range temporal correlation between two similar events, the first touch and the second touch, in order to identify the proper modifier as 2 times. In Fig. 10, we show the sampled images from the CL2AD dataset. With a static image, it is difficult to infer the action. But it is even more difficult to infer the action modifier.

**Results**: The result obtained by testing the MSTRNN, MSTNN, and LRCN on CL2AD, which has a higher level of compositionality than the previously introduced datasets (3ACWD, CL1AD), is shown in Table V. The MSTRNN exhibited higher categorization performance in all categories (ADO, action, and modifier) than the model without recurrent structures (MSTNN) and the model with LSTM layers to process temporal information (LRCN). We again find that the recurrent structure, the capacity to extract spatio-temporal features simultaneously, and the adequate multiscale temporal constraints of the proposed model help it during action recognition tasks.

TABLE V
RECOGNITION ACCURACIES ON CL2AD IN PERCENTAGES

| | | MSTRNN | LRCN | MSTNN |
|---|---|---|---|---|
| ADO | Accuracy | **92.86±2.12** | 77.98±3.52 | 88.93±2.24 |
| | Chance Level | 25.00 | | |
| Action | Accuracy | **81.43±3.26** | 78.10±4.12 | 71.67±2.82 |
| | Chance Level | 25.00 | | |
| Modifier | Accuracy | **80.36±2.70** | 75.24±3.15 | 65.12±2.18 |
| | Chance Level | 16.67 | | |

The LRCN showed the second best performances, except for the ADO category. The MSTNN showed higher performance than the LRCN during the ADO categorization task. This result may be due to the fact that the LRCN extracts spatial and temporal features separately, while the MSTNN extracts spatio-temporal features simultaneously.

## IV. Conclusions and Discussions

We proposed a MSTRNN model for action recognition which extracts spatio-temporal features simultaneously using multiscale spatio-temporal constraints imposed on the neural activities in different layers. The MSTRNN, MSTNN and LRCN were compared by using the 3ACWD, CL1AD, and CL2AD, which have different compositionality levels. All experimental results show that the MSTRNN outperforms the baselines (MSTNN and LRCN) despite its fewer parameters than in the baseline models. This result shows that the recurrent structure, spatio-temporal constraints, and the simultaneous extraction of the spatio-temporal information of the MSTRNN are helpful for the recognition of actions. From experiments with 3ACWD and CL1AD, comparative analytic results on the neural activation sequences of the models (MSTRNN, MSTNN, and LRCN) were obtained. The results suggest that the MSTRNN can develop more enhanced categorical memories by which the compositional categorization of visually grounded data can be achieved more effectively as compared to that by the MSTNN and LRCN. This is consistent with biological evidence showing that a cat can identify past and current visual inputs by utilizing the recurrent connectivity present in its visual cortex [14], [15] and that the spatio-temporal receptive field grows as the layer goes up in a mammalian cortex [9], [10]. The results are also consistent with the principle of downward causation that argues spatio-temporal hierarchy, which is considered to be beneficial for the formation of categorical memories, can be self-organized by the setting of global spatio-temporal constraints [11], [12].

Although the MSTRNN showed better performance than the other models in the comparison, its action recognition capacity is not high enough for a practical use. One of the main reasons for such degeneracy may be overfitting. In future work, it may be possible to alleviate this overfitting problem with recently developed deep learning regularization techniques, including the dropout technique for recurrent connections based on variational inference [36] and layer normalization [37].

In this study, we used a recurrent structure with CNNs to categorize human actions. However, a recent study of hybrid networks of CNNs and RNNs has shown good performance in recognizing objects [38]. It was reported recurrent convolutional neural networks (RCNNs) [38] have enhanced object (static image) recognition capacities compared to feed forward CNNs due to their recurrent convolutional structures. The recurrent convolutions of a RCNN enable it to use context information for a static image recognition task. This mechanism is similar to how V1 neurons change their manner of interpreting input received from their receptive fields (RFs), as influenced by the spatial context near their RFs [39]. In this context, the proposed model has the potential to perform well on object recognition tasks due to its recurrent connectivity.

With the development of the spatio-temporal hierarchy using multiscale spatio-temporal constraints, the proposed model may be able to exploit more context information for the recognition of objects in videos compared to a model without temporal constraints (RCNNs). Moreover, the capacity to recognize objects in videos could be used for robotic vision. Therefore, our future research may focus on applying our model to recognize objects from dynamic images. With regard to video generation tasks, a model that shares similar features to our model was successfully applied to the domain of video generation [40]. The predictive MSTRNN (P-MSTRNN) [40] has shown that the mechanisms of the MSTRNN are helpful during the generation of videos. Specifically, the spatio-temporal hierarchy developed by spatio-temporal constraints can be useful during the generation of videos. In future work, it would be interesting to incorporate the P-MSTRNN into the model proposed here to enhance its action recognition capability.

Recently, Zhang et al. [41] found that a cortical projection is made from the visual cortex (cingulate region) of a mouse in a top-down manner to V1 neurons to increase their sensitivity to visual input. Inspired by this biological finding, we are interested in adding a top-down prediction pathway (the P-MSTRNN structure may be used) with the goal of generating attention patches where the model would obtain input images that are relevant to action recognition tasks. The future model will be more computationally effective and would show enhanced performance compared to our currently proposed model because it could disregard areas of the images that are not relevant to the given task. This may eventually give our future model the potential to scale up to larger datasets with cluttered scenes [42].

## V. Appendix

Table VI shows the result of the sensitivity analysis made on the time constants of the MSTRNN when the input is CL2AD. The analysis was conducted with CL2AD because it is the most complex one regarding compositionality among the three datasets (i.e., 3ACWD, CL1AD, and CL2AD) used in this work. In the table, $\tau_1$ and $\tau_2$ refer to the time constants assigned to the first and second context layers of the MSTRNN, respectively. The result shows that the action recognition performance of the MSTRNN is highest when $\tau_1 = 2$ and $\tau_2 = 100$. For the modifier recognition task, $\tau_1 = 2$ and $\tau_2 = 150$ showed the best performance. But, the performance was only slightly higher than when $\tau_1 = 2$ and $\tau_2 = 100$ by 0.59% (mean value). Regarding the standard errors obtained when the settings were $\tau_1 = 2$, $\tau_2 = 100$ and $\tau_1 = 2$, $\tau_2 = 150$, this difference is not significant and can be neglected. Lastly, for the ADO recognition, the performance was highest for the setting of $\tau_1 = 100$, $\tau_2 = 50$. This result may be because the ADO recognition does not require much spatio-temporal information processing capacity as in the case of recognizing actions and modifiers. In fact, the setting of $\tau_1 = 100$, $\tau_2 = 50$ made the MSTRNN model perform lower than the MSTRNN with the setting of $\tau_1 = 2$, $\tau_2 = 100$ in the action and modifier recognition tasks. Since the primary task of the MSTRNN is to recognize actions, it can be concluded that the setting of $\tau_1 = 2$, $\tau_2 = 100$ is best for the MSTRNN model used in the experiments and within the searched pairs of time constants. This result is in accordance with the neuroscientific evidence that the temporal receptive windows increase as the layer goes up in the human brain [10].

TABLE VI
Sensitivity analysis of the time constants in the MSTRNN

| Category | $\tau_2$<br>$\tau_1$ | 50 | 100 | 150 |
|---|---|---|---|---|
| ADO | 2 | 90.12±4.14 | 92.86±2.12 | 88.69±5.79 |
| | 50 | 93.45±3.83 | 90.48±6.55 | 93.21±2.72 |
| | 100 | **93.93±3.05** | 92.98±4.17 | 91.07±4.91 |
| Action | 2 | 79.64±3.69 | **81.43±3.26** | 79.40±3.19 |
| | 50 | 80.24±3.13 | 81.31±3.22 | 80.36±2.65 |
| | 100 | 79.64±3.06 | 79.52±3.73 | 79.64±4.12 |
| Modifier | 2 | 77.86±2.15 | 80.36±2.70 | **80.95±2.64** |
| | 50 | 78.21±2.55 | 79.05±2.01 | 79.76±2.30 |
| | 100 | 77.50±2.20 | 77.74±3.16 | 76.79±3.10 |

## Acknowledgment

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

## References

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2–9, 2009.

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June-2015, 2015, pp. 1–9.

[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, “ImageNet Large Scale Visual Recognition Challenge,” *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[5] S. Ji, W. Xu, M. Yang, and K. Yu, “3d convolutional neural networks for human action recognition,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[6] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, T. Darrell, and K. Saenko, “Long-term recurrent convolutional networks for visual recognition and description,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June-2015, 2015, pp. 2625–2634.

[7] K. Simonyan and A. Zisserman, “Two-stream convolutional networks for action recognition in videos,” in *Advances in neural information processing systems*, 2014, pp. 568–576.

[8] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, “Learning Precise Timing with LSTM Recurrent Networks,” *Journal of Machine Learning Research*, vol. 3, no. 1, pp. 115–143, 2002.

[9] D. J. Felleman and D. C. Van Essen, “Distributed hierarchical processing in the primate cerebral cortex,” *Cerebral Cortex*, vol. 1, no. 1, pp. 1–47, 1991.

[10] U. Hasson, E. Yang, I. Vallines, D. J. Heeger, and N. Rubin, “A hierarchy of temporal receptive windows in human cortex.” *The Journal of neuroscience*, vol. 28, no. 10, pp. 2539–50, 2008.

[11] D. T. Campbell, “downward causationin hierarchically organised biological systems,” in *Studies in the Philosophy of Biology*. Macmillan Education UK, 1974, pp. 179–186.

[12] D. S. Bassett and M. S. Gazzaniga, "Understanding complexity in the human brain," *Trends in cognitive sciences*, vol. 15, no. 5, pp. 200–209, 2011.

[13] M. Jung, J. Hwang, and J. Tani, "Self-organization of spatio-temporal hierarchy via learning of dynamic visual image patterns on action sequences," *PloS one*, vol. 10, no. 7, p. e0131214, 2015.

[14] D. V. Buonomano and W. Maass, "State-dependent computations: spatiotemporal processing in cortical networks," *Nature Reviews Neuroscience*, vol. 10, no. 2, pp. 113–125, 2009.

[15] D. Nikolić, S. Häusler, W. Singer, and W. Maass, "Distributed fading memory for stimulus properties in the primary visual cortex," *PLoS biology*, vol. 7, no. 12, p. e1000260, 2009.

[16] M. A. Arbib, "Perceptual structures and distributed motor control," *Handbook of Physiology - The Nervous System II. Motor Control*, pp. 1449–1480, 1981.

[17] J. Tani, "Self-organization and compositionality in cognitive brains: A neurorobotics study," *Proceedings of the IEEE*, vol. 102, no. 4, pp. 586–605, 2014.

[18] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2247–2253, 2007.

[19] Y. Yamashita and J. Tani, "Emergence of functional hierarchy in a multiple timescale neural network model: a humanoid robot experiment," *PLoS Comput Biol*, vol. 4, no. 11, p. e1000220, 2008.

[20] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[21] T. Cooijmans, N. Ballas, C. Laurent, Ç. Gülçehre, and A. Courville, "Recurrent batch normalization," *arXiv preprint arXiv:1603.09025*, 2016.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[23] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[24] M. I. Jordan, "Attractor dynamics and parallellism in a connectionist sequential machine," 1986.

[25] Y. LeCun *et al.*, "Generalization and network design strategies," *Connectionism in perspective*, pp. 143–155, 1989.

[26] D. E. Rumelhart, J. L. McClelland, P. R. Group *et al.*, *Parallel distributed processing*. IEEE, 1988, vol. 1.

[27] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

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

[29] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[30] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015, pp. 2342–2350.

[31] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[32] H. Hotelling, "Analysis of a complex of statistical variables into principal components." *Journal of educational psychology*, vol. 24, no. 6, p. 417, 1933.

[33] X.-Y. Zhang, F. Yin, Y.-M. Zhang, C.-L. Liu, and Y. Bengio, "Drawing and recognizing chinese characters with recurrent neural network," *arXiv preprint arXiv:1606.06539*, 2016.

[34] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," in *Advances in neural information processing systems*, 2015, pp. 2980–2988.

[35] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[36] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 1019–1027.

[37] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[38] M. Liang and X. Hu, "Recurrent convolutional neural network for object recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3367–3375.

[39] P. Series, J. Lorenceau, and Y. Frégnac, "The silent surround of v1 receptive fields: theory and experiments," *Journal of physiology-Paris*, vol. 97, no. 4, pp. 453–474, 2003.

[40] M. Choi and J. Tani, "Predictive coding for dynamic vision: Development of functional hierarchy in a multiple spatio-temporal scales rnn model," *arXiv preprint arXiv:1606.01672*, 2016.

[41] S. Zhang, M. Xu, T. Kamigaki, J. P. H. Do, W.-C. Chang, S. Jenvay, K. Miyamichi, L. Luo, and Y. Dan, "Long-range and local circuits for top-down modulation of visual cortex processing," *Science*, vol. 345, no. 6197, pp. 660–665, 2014.

[42] V. Mnih, N. Heess, A. Graves *et al.*, "Recurrent models of visual attention," in *Advances in neural information processing systems*, 2014, pp. 2204–2212.

**Haanvid Lee** received the B.S. degree in electrical and electronic engineering from Yonsei University, Seoul, Republic of Korea, in 2015, and the M.S. degree in electrical engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Republic of Korea, in 2017.

He is currently a Ph.D. candidate in the School of Computing at KAIST. His current research interests include computer vision, reinforcement learning, and inverse reinforcement learning.

**Minju Jung** received the B.E. degree in electronic engineering from Hanyang University, Seoul, Republic of Korea, in 2013.

He is currently pursuing the Ph.D. degree in electrical engineering at Korea Advanced Institute of Science and Technology, Daejeon, Republic of Korea. His current research interests include recurrent neural networks and computer vision.

**Jun Tani** received the B.S. degree in mechanical engineering from Waseda University, Tokyo, Japan, the M.S. degrees in electrical engineering and mechanical engineering from the University of Michigan, Ann Arbor, MI, USA, and the D.Eng. degree from Sophia University, Tokyo.

He started his research career with the Sony Laboratory, Tokyo, Japan, in 1990. He had been a Team Leader of the Laboratory for Behavior and Dynamic Cognition, RIKEN Brain Science Institute, Saitama, Japan, for 12 years until 2012. He was a Visiting Associate Professor with the University of Tokyo, Tokyo, from 1997 to 2002. Also, he was a Full Professor with the Electrical Engineering Department, Korea Advanced Institute of Science and Technology, Daejeon, Republic of Korea, from 2012 to 2017. He is currently a Full Professor with the Okinawa Institute of Science and Technology, Okinawa, Japan. His current research interests include neuroscience, psychology, phenomenology, complex adaptive systems, and robotics.