IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 12 September 2025, accepted 17 October 2025, date of publication 21 October 2025, date of current version 30 October 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3624029

RESEARCH ARTICLE

SSTRAC: Skeleton-Based Dual-Stream Spatio-Temporal Transformer for Repetitive Action Counting in Videos

JUNGJUN LIM[1], DONGHOON KANG[2], (Member, IEEE), KANGHYUN RYU[2], (Member, IEEE), AND JE HYEONG HONG[3], (Member, IEEE)

[1]Department of Electrical Engineering, Korea University, Seoul 02841, South Korea
[2]Intelligence and Interaction Research Center, Korea Institute of Science and Technology (KIST), Seoul 02792, South Korea
[3]Department of Electronic Engineering, Hanyang University, Seoul 04763, South Korea

Corresponding author: Donghoon Kang (kimbab.moowoo@gmail.com)

This work was supported in part by the Korea Institute of Science and Technology (KIST) under Grant 2V09141, Grant 2V09232 (KIST K-DARPA Project), and Grant 2E33841 (KIST Institutional Project); and in part by the National Research Foundation of Korea (NRF) Grant funded by the Korea government (MSIT) under Grant RS-2022-NR070020.

ABSTRACT Most existing approaches to predicting the number of repetitive actions in videos focus on improving model accuracy, but overlook important issues such as robustness to changes in human body size and and occlusion of human body parts in videos. To achieve robustness to changes in human size and and occlusion of human body in videos, we propose a novel network, Skeleton-based dual-stream Spatio-temporal Transformer for Repetitive Action Counting (SSTRAC) using videos, which reconstructs defective human skeletons as a preprocessing step, and then encodes the spatial and temporal information of repetitive actions into the per-frame embeddings through the dual-stream spatio-temporal transformer. To capture both high and low frequency actions in short and long videos, the per-frame embeddings are abstracted in the form of a multi-scale self-attention matrix. In the final step, the period predictor estimates a density map, which provides the number of repetitive actions in each video. We performed extensive experiments by comparing the proposed model with other recent state-of-the art models. The experimental results demonstrate the superiority of our model in terms of robustness to changes in human size and occlusion of human body parts in videos. Codes and models are available at https://github.com/imjjun/SSTRAC_public

INDEX TERMS Repetitive action counting, skeleton, human size, occlusion, spatio-temporal, multi-scale, density map.

I. INTRODUCTION

In recent years, various methods [1], [2], [3], [4], [5] for repetitive action counting (RAC) in videos automatically have been actively studied. However, most of the existing methods for RAC which directly use videos focus on enhancing the accuracy of models, but overlook the important issues such as robustness to change in human size and occlusion of human body parts in videos. People and cameras can move freely, so the size of the people's images in videos can vary. Because the number of dataset for RAC is small, the performance of models can be greatly affected by changes in human size. Thus, the performance of existing models for RAC which directly use videos can be degraded when there is change in human size and occlusion in videos.

In this study, we investigate the relatively less explored but practically significant issue: how to develop a model for RAC that achieves **robustness to changes in human body size and occlusion of human body parts** in videos. To address such issue, we introduce a novel approach, Skeleton-based dual-stream Spatio-temporal Transformer for

The associate editor coordinating the review of this manuscript and approving it for publication was Laxmisha Rai.

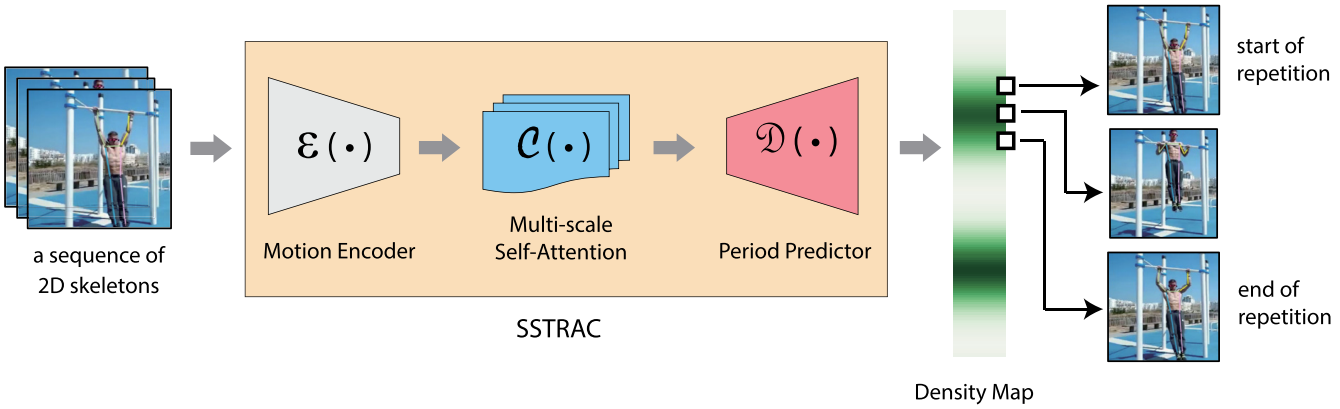


FIGURE 1. SSTRAC architecture. We propose the Skeleton-based dual-stream Spatio-temporal Transformer for Repetitive Action Counting (SSTRAC) in videos. SSTRAC consists of three main components: (i) motion encoder, (ii) a component that obtains a multi-scale self-attention matrix, and (iii) period predictor. The SSTRAC takes a sequence of 2D skeletons as input, and then predicts a dense map and the count of repetitive action in a video.

Repetitive Action Counting (**SSTRAC**) in videos (see Fig. 1). To the authors' knowledge, there is little research for RAC to achieve the robustness to changes in human body size and occlusion of human body parts.

Our work is inspired by the recent RAC model, TransRAC [2], and human motion representation model, MotionBERT [6], but has the following fundamental differences: (i) Unlike TransRAC [2] which directly uses videos, our model only processes **human motion data** as input in the form of skeleton sequences. Although our model may not count the repetitive action of non-human objects (e.g., pets, curtains) in videos, it focuses more on human action. (ii) In our design, we use Dual-stream spatio-temporal transFormer (DstFormer) in MotionBERT [6] as a component of our model (*i.e.*, motion encoder in Fig. 1) to learn human motion representations. In MotionBERT [6], DstFormer is trained by accomplishing the 2D-to-3D lifting task, whereas in our case it is trained as a component for our model by encoding rich spatio-temporal information of repetitive actions. Moreover, unlike MotionBERT [6], our model does not require any additional fine-tuning process.

To predict the number of repetitive actions in the video, several methods have been studied which use **skeletons** as input [7], [8]. This is because the skeleton can be more robust to camera view changes and background occlusion than RGB image data [9], [10]. The size of people in the video may change as people or the camera move. When using 2D pose estimator, the performance of the model for RAC can be robust to changes in the size of the person in the video. This is because the 2D pose estimator normalizes the size of human skeleton. Furthermore, the model that uses skeleton sequences as input can be constructed lightweight because the dimension of skeleton data is lower than RGB image data [11], [12], [13].

However, except for PoseRAC [14], most existing models [7], [8] that use skeleton sequences as input, do not predict density maps as a fine-grained representation to estimate the number of repetitions in the video. In this case, the models may incorrectly predict the start and end video frames of the repetitive actions, but they can correctly predict the number of repetitions by chance. To reduce these errors, it is helpful to predict the **density map** [2] as well as the total number of repetitive actions in the video. As shown in the rightmost part of Fig. 1, the dark areas of the density map indicate that a single repetition has occurred. Thus, density maps may be helpful to interpret the performance of models correctly.

Compared to PoseRAC [14], our model has the following fundamental differences: (i) PoseRAC requires more information such as salient poses to train than our model. Therefore, to train PoseRAC, we need to annotate salient poses to the existing RepCount-A dataset [2], which labels the start and end video frames of repetitions in videos. For this reason, comparing directly our model to PoseRAC may not be entirely fair. (ii) Although PoseRAC requires additional information about salient poses during training, this can result in the lower accuracy of the model. This is because the PoseRAC relies heavily on too small number (*i.e.* only two) of salient poses to infer the number of repetitive actions in a single video.

To achieve both model robustness and interpretability, the proposed SSTRAC model shown in Fig. 1 takes a sequence of skeletons as input and predicts a density map as a fine-grained representation. Our model can address human size changes and occlusion issues caused by human and camera movements by adopting an occlusion-robust 2D pose estimator [15]. The pose estimator generates 2D skeletons whose sizes are **normalized**. This benefit may result from the design of our model, using a sequence of 2D skeletons as input.

As shown in Fig. 1, the SSTRAC model consists of three main components: (i) motion encoder, (ii) a component for obtaining a multi-scale self-attention matrix, and (iii) period predictor. In motion encoder the repetitive human actions in 2D skeleton sequences are encoded into

per-frame embeddings through dual-stream spatio-temporal transformer (see Fig. 2). The per-frame embeddings are then abstracted to the multi-scale self-attention matrix as shown in Fig. 4. The multi-scale self-attention matrix can be obtained by computing three different self-attention matrices to capture repetitive actions with different period lengths. In the final step, the period predictor predicts the density map, which enables to compute the total number of repetitive actions in the video (see the right part of Fig. 4).

The main contributions of this paper are summarized as follows:

- We propose a novel network, Skeleton-based dual-stream Spatio-temporal Transformer for Repetitive Action Counting (SSTRAC), which consists of three main components: (i) motion encoder, (ii) a component for obtaining a multi-scale self-attention matrix, and (iii) period predictor. One of the attractive features of our model is that it can achieve model robustness and interpretability by using skeletons as input and density map as a fine-grained representation.
- In a motion encoder, rich spatial and temporal information about human action in a skeletal sequence are encoded in the form of a dual-stream. To capture both high and low frequency action in short and long videos, a multi-scale self-attention matrix is obtained. In the final step, the period predictor estimates a density map, which provides the number of repetitive actions in each video.
- We conducted extensive experiments by comparing the proposed model with other recent state-of-the art models. Experimental results show that the proposed model is superior to other state-of-the art models in terms of robustness against changes in human size and occlusion of human body parts in videos.

II. RELATED WORK

Existing models for RAC in videos can be classified into two categories according to the type of input data: (i) RGB image-based models and (ii) skeleton-based models. RGB image-based models directly use images in video frames as input. In contrast, skeleton-based models utilize a sequence of human skeletons as input, which are extracted from the video using a human pose estimator. The SSTRAC model in this paper belongs to the skeleton-based models for RAC.

A. RGB IMAGE-BASED MODELS FOR REPETITIVE ACTION COUNTING

Zhang et al. [16] proposed a context-aware and scale-insensitive model to estimate the count of repetitive action in videos. However, it requires an exhaustive search to obtain an initial guess about the length of double-cycle in the video. RepNet [1], a class-agnostic model, uses a temporal self-similarity matrix as an intermediate layer, and then predicts both the period and count of repetitions in relatively short videos (e.g., $0.4-30$s). Li et al. [3] presented a model named Hybrid Temporal Relation Modeling Network (HTRM-Net) to enhance the representation of temporal self-similarity matrices through the exploitation of hybrid temporal relation modeling. Zhang et al. [4] proposed a framework to address the issue of period inconsistency and motion interruption in RAC. To overcome the period inconsistency challenge, they developed a boundary-aware encoder which compresses the temporal resolution of features to capture cycles of varying temporal lengths. To address the issue of motion interruption, they predicted the probability of each video frame falling within the motion cycle to generate actionness scores. Li and Xu [17] proposed a two-branch framework, *i.e.*, RGB branch and flow branch to improve the foreground motion feature learning. Although the aforementioned studies have made significant progress in RAC research, their performance may be degraded if videos contain frequent changes in body size or occluded body parts.

B. SKELETON-BASED MODELS FOR REPETITIVE ACTION COUNTING

In human action recognition research using videos, skeleton-based approaches have been actively studied because skeleton data is more robust to changes in lighting, camera view, background occlusion, etc. than RGB image data. Additionally, skeleton data has lower dimensionality than RGB images, allowing for designing lightweight models [9], [10]. However, when narrowing down to the task of RAC in videos, there are not many studies using skeletons. PoseRAC [14], which uses a sequence of skeletons for RAC in the video as input to the model, requires two salient human poses.

C. DATASETS FOR REPETITIVE ACTION COUNTING

Recently, considerable efforts have been made to collect video datasets for various action understanding tasks. As a result of these efforts, numerous datasets for action understanding from videos have been built and made publicly available [18], [19], [20]. In contrast, there are very few publicly available datasets for the task of RAC in videos. To address the problem of insufficient video datasets for RAC, various data augmentation methods have been studied [1]. Among the datasets for RAC that have been released to the public so far, the most representative ones include QUVA repetition dataset [21], Countix [1], UCFRep [16], and RepCount-A [2].

III. PROPOSED MODEL: SSTRAC

In this section, we introduce our model, **SSTRAC** for counting repetitive human action in a video. Our model aims to predict the count of repetitive action in videos while simultaneously handling changes in human body size and the occlusion of human body parts. As illustrated in Fig. 1, the SSTRAC consists of three major components: (i) motion encoder, (ii) a component for obtaining

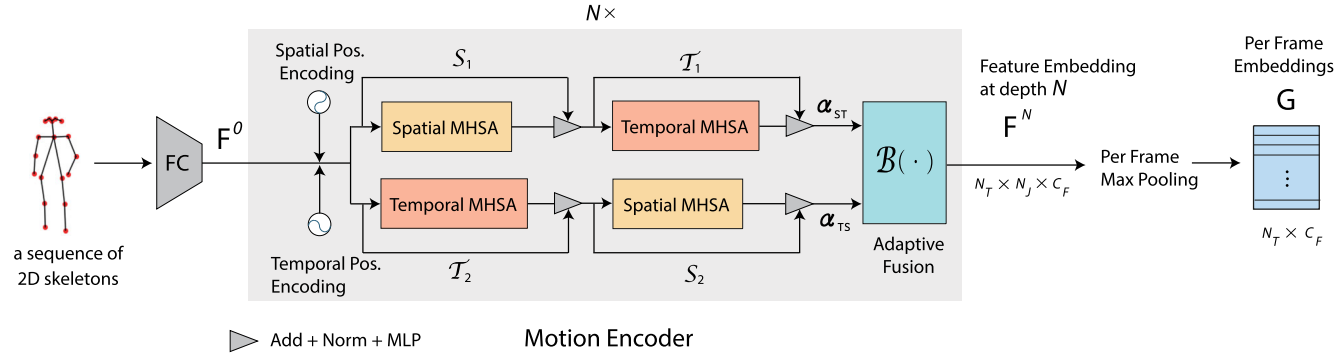


FIGURE 2. Motion encoder.

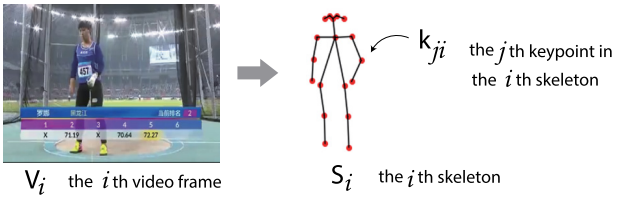


FIGURE 3. Mathematical notations: Given the i-th video frame, V_i, a skeleton $\mathbf{S}_i$ can be extracted by using 2D pose estimator. The j-th row of $\mathbf{S}_i$, denoted $\mathbf{k}_{ji}$ represents the j-th keypoint in the i-th skeleton.

a multi-scale self-attention matrix, and (iii) period predictor. Below we describe the three main components in more detail.

A. SQUENCE OF SKELETONS

Given N_T consecutive video frames, $\mathbf{V} = [V_1, V_2, \ldots, V_{N_T}]$, our model, the SSTRAC, first extracts a sequence of *skeletons*, $\mathbf{S} = [\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_{N_T}] \in \mathbb{R}^{N_T \times N_J \times N_D}$ using a human 2D pose estimator [15] that is robust to occlusion of human body parts in the image. Here N_T is the number of video frames, *i.e.*, the sequence length, and N_J denotes the number of *keypoints* in the skeleton. In our design, $N_T = 256$ and $N_J = 17$. As shown in Fig. 3, keypoints of the skeleton are usually assigned to human body joints. N_D denotes the coordinate dimension of each keypoint. Thus, $N_D = 2$ (for 2D keypoints).

The j-th row of the i-th skeleton matrix, $\mathbf{S}_i \in \mathbb{R}^{N_J \times N_D} (i = 1 \ldots N_T)$ extracted at the i-th video frame V_i, is denoted $\mathbf{k}_{ji} \in \mathbb{R}^{1\times 2}$, $(j = 1 \ldots N_J)$ as follows:

$$\mathbf{S}_i = \begin{bmatrix} -\mathbf{k}_{1i}- \\ \vdots \\ -\mathbf{k}_{N_J\ i}- \end{bmatrix} \tag{1}$$

where $\mathbf{k}_{ji}$ represents the 2D coordinate vector of the j-th keypoint in the i-th skeleton, $\mathbf{S}_i$ (see Fig. 3).

B. MOTION ENCODER

The motion encoder shown in Fig. 2 adopts the DstFormer in MotionBERT [6], but has the following differences: The motion encoder in Fig. 2 is trained as a component for the SSTRAC model by capturing spatio-temporal *repetitive* human *motion features*. In contrast, the DstFormer in MotionBERT [6] is trained by accomplishing the 2D-to-3D lifting task.

1) SPATIAL AND TEMPORAL POSITIONAL ENCODINGS

As illustrated in Fig. 2, a sequence of skeletons, $\mathbf{S} = [\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_T]$ is first fed to a fully connected layer (FC), and then projected into the high-dimensional features, denoted $\mathbf{F}^0 \in \mathbb{R}^{N_T \times N_J \times C_F}$. Here, C_F is the feature size. $C_F = 512$. We call $\mathbf{F}^0$ the feature embedding at depth 0. Then we add learnable spatial positional encoding $\mathbf{P}_{spatial} \in \mathbb{R}^{1\times N_J \times C_F}$ and temporal positional encoding $\mathbf{P}_{temporal} \in \mathbb{R}^{N_T \times 1 \times C_F}$ to $\mathbf{F}^0$.

2) DUAL-STREAM SPATIO-TEMPORAL TRANSFORMER

Motion encoder in Fig. 2 has a form of dual-stream which contains spatial multi-head self-attention (Spatial MHSA) and temporal multi-head self-attention (Temporal MHSA). Fig. 2 shows the dual-stream architecture of the motion encoder. In the upper part of Fig. 2, the spatial MHSA comes first, followed by the temporal MHSA. Similarly, the lower part of Fig. 2 has the temporal MHSA followed by the spatial MHSA. The output feature of two streams are fused using adaptive fusion weights, denoted $\alpha_{ST}, \alpha_{TS} \in \mathbb{R}^{N\times N_T \times N_J}$ where N is the depth of DstFormer. The feature embedding $\mathbf{F}^i$ at depth $i, (i \in 1, \ldots, N)$ is computed by iterating the following equation:

$$\mathbf{F}^i = \alpha^i_{ST} \circ \mathcal{T}^i_1(\mathcal{S}^i_1(\mathbf{F}^{i-1})) + \alpha^i_{TS} \circ \mathcal{S}^i_2(\mathcal{T}^i_2(\mathbf{F}^{i-1})),$$

where $\circ$ represents element-wise multiplication. Here $\mathcal{S}^i_1$ and $\mathcal{S}^i_2$ denote the spatial MHSA shown in Fig. 2. Likewise $\mathcal{T}^i_1$ and $\mathcal{T}^i_2$ are the temporal MHSA (see Fig. 2). The adaptive fusion weights, α_{ST}, α_{TS} are calculated by

$$\alpha^i_{ST}, \alpha^i_{TS} = \text{softmax}(\mathcal{W}([\mathcal{T}^i_1(\mathcal{S}^i_1(\mathbf{F}^{i-1}))])), \mathcal{S}^i_2(\mathcal{T}^i_2(\mathbf{F}^{i-1}))])),$$

where $\mathcal{W}$ is a learnable linear transformation and $[,]$ represents a concatenation operation.

Unlike DstFormer in [6], the motion encoder of our model does not require additional FC layers after adaptive fusion

(see Fig. 2). For more details on the motion encoder in Fig. 2, refer to the DstFormer in [6]. As shown in Fig. 2, the output of motion encoder is per-frame embeddings, $\mathbf{G}$, where its size is $N_T \times C_F$.

C. MULTI-SCALE SELF-ATTENTION MATRIX

To capture high and low frequency repetitive action in short and long videos, our model uses the self-attention in TransRAC [2]. While TransRAC [2] obtains a self-attention for per-frame embeddings encoded from a sequence of video frames, in our work we compute self-attention for per-frame embeddings, $\mathbf{G} \in \mathbb{R}^{N_T \times C_F}$ encoded from a *skeleton* sequence.

Let $\mathbf{g}_i \in \mathbb{R}^{1 \times C_F}, (i = 1, \ldots, N_T)$ denote the i-th row vector of $\mathbf{G}$ (see the leftmost part of Fig. 4) as follows:

$$\mathbf{G} = \begin{bmatrix} -\mathbf{g}_1- \\ \vdots \\ -\mathbf{g}_{N_T}- \end{bmatrix}.$$

From $\mathbf{G}$, we construct three scale subsequences, $\mathbf{X}_1, \mathbf{X}_4$, and $\mathbf{X}_8$, which are defined as

$$\begin{aligned} \mathbf{X}_1 &= [\{\mathbf{g}_1\}; \{\mathbf{g}_2\}; \ldots] \\ \mathbf{X}_4 &= [\{\mathbf{g}_1, \ldots, \mathbf{g}_4\}; \{\mathbf{g}_5, \ldots, \mathbf{g}_8\}; \ldots] \\ \mathbf{X}_8 &= [\{\mathbf{g}_1, \ldots, \mathbf{g}_8\}; \{\mathbf{g}_9, \ldots, \mathbf{g}_{16}\}; \ldots] \end{aligned} \tag{2}$$

Three types of skeletal sequences with different scales can provide useful information for capturing repetitive human action with different period lengths. In Eq.(2), the first row of $\mathbf{X}_4$ is $\{\mathbf{g}_1, \ldots, \mathbf{g}_4\}$ which represents the concatenation of row vectors, $\mathbf{g}_1, \ldots, \mathbf{g}_4$ along the column-wise direction. Each row of $\mathbf{X}_1$ and $\mathbf{X}_8$ in Eq.(2) is defined similarly to $\mathbf{X}_4$. Here $\mathbf{X}_k, (k = 1, 4, 8)$ is constructed using a sliding window with a step size k. The shape of matrices, $\mathbf{X}_1, \mathbf{X}_4$, and $\mathbf{X}_8$ are $(N_T,\ 1 \times C_F), (\frac{N_T}{4},\ 4 \times C_F), (\frac{N_T}{8},\ 8 \times C_F)$, respectively.

Then, we can obtain three different self-attention matrices, $\mathbf{C}_1, \mathbf{C}_4$, and $\mathbf{C}_8$ as follows: $\mathbf{C}_k = \psi(\mathbf{X}_k), (k = 1, 4, 8)$ where $\psi(\cdot)$ is a multi-head self-attention (MSA) [22]. As shown in Fig. 4, $\psi(\cdot)$ computes a multi-scale self-attention matrix by encoding temporal correlations at different scales. The shape of each $\mathbf{C}_k$ is (N_T, N_T, h) which implies that its size is $N_T \times N_T \times h$. Here h is the number of heads (see Fig. 4) and $h = 4$ in our design. By concatenating $\mathbf{C}_1, \mathbf{C}_4$ and $\mathbf{C}_8$ along the channel-wise direction, we can construct a multi-head self attention matrix, $\mathbf{H}$ as follows: $\mathbf{H} = \{\mathbf{C}_1, \mathbf{C}_4, \mathbf{C}_8\}$ whose shape is $(N_T, N_T, 3 \times h)$.

D. PERIOD PREDICTOR ESTIMATING DENSITY MAP

In this section, we introduce a period predictor, the third component of the SSTRAC model (see the right part of Figs. 1 and 4). The goal of period predictor is to estimate a dense map that can be used to count repetitive action in a video. To achieve this goal, we use the period predictor in TransRAC [2].

1) DENSITY MAP

In this paper, the ground truth of a density map is called the *target* density map. Target density map can be represented as a column vector, $\mathbf{m} = (m_1, \ldots, m_{N_T}) \in \mathbb{R}^{N_T}$. To understand what density maps mean, it is helpful to figure out how the target density map is generated from video datasets. In Section V-B, we explain how to generate target density maps from video datasets. The meaning of the density map can be explained as follows: The i-th component m_i of the target density map $\mathbf{m}$ represents the probability of a single count in the i-th video frame during a repetitive action. Thus, we can compute

$$c = \sum_{i=1}^{N_T} m_i \tag{3}$$

where $c \in \mathbb{R}$ is the number of repetitive actions in the video.

2) PERIOD PREDICTOR

Now let us briefly explain how the period predictor [2] is constructed. As shown in the right part of Fig. 4, we first apply a ConvBlock (consisting of thirty two 3×3 convolutional layers, batch normalization and ReLU function) to the multi-head self attention matrix, $\mathbf{H}$.

To project the features into a $N_T \times 512$-dimensional space, we applied a linear layer to ConvBlock ($\mathbf{H}$). Then we use the transformer encoder to encode the count of repetitive action in the skeletal sequences, taking the per-frame features as an input. The output of transformer encoder whose shape is $(N_T, 512)$, is then transferred to FC layer, which predicts the density map (see the rightmost part of Fig. 4). The predicted density map is represented as a column vector, $\hat{\mathbf{m}} = (\hat{m}_1, \ldots, \hat{m}_{N_T}) \in \mathbb{R}^{N_T}$.

From the estimated density map $\hat{\mathbf{m}}_i$, we can infer the total count, $\hat{c} \in \mathbb{R}$, of repetitive action in the video as

$$\hat{c} = \sum_{i=1}^{N_T} \hat{m}_i \tag{4}$$

which is the final result of our model.

E. TRAINING LOSS

The total loss for training our model has the following form

$$\mathcal{L} = \lambda_1 \mathcal{L}_{Map} + \lambda_2 \mathcal{L}_{Count} \tag{5}$$

where $\mathcal{L}_{Map}$ is the mean square error (MSE) loss of the predicted density map and the target density map, which is formulated as follows:

$$\mathcal{L}_{Map} = \frac{1}{N_V} \sum_{k=1}^{N_V} \mathcal{L}_M^k(\hat{\mathbf{m}}^{(k)}, \mathbf{m}^{(k)}) \tag{6}$$

where N_V is the total number of videos. The loss function $\mathcal{L}_M^k(\hat{\mathbf{m}}^{(k)}, \mathbf{m}^{(k)})$ in Eq.(6) is defined as

$$\mathcal{L}_M^k(\hat{\mathbf{m}}^{(k)}, \mathbf{m}^{(k)}) = \frac{1}{N_{T_k}} \|\hat{\mathbf{m}}^{(k)} - \mathbf{m}^{(k)}\|^2 \tag{7}$$

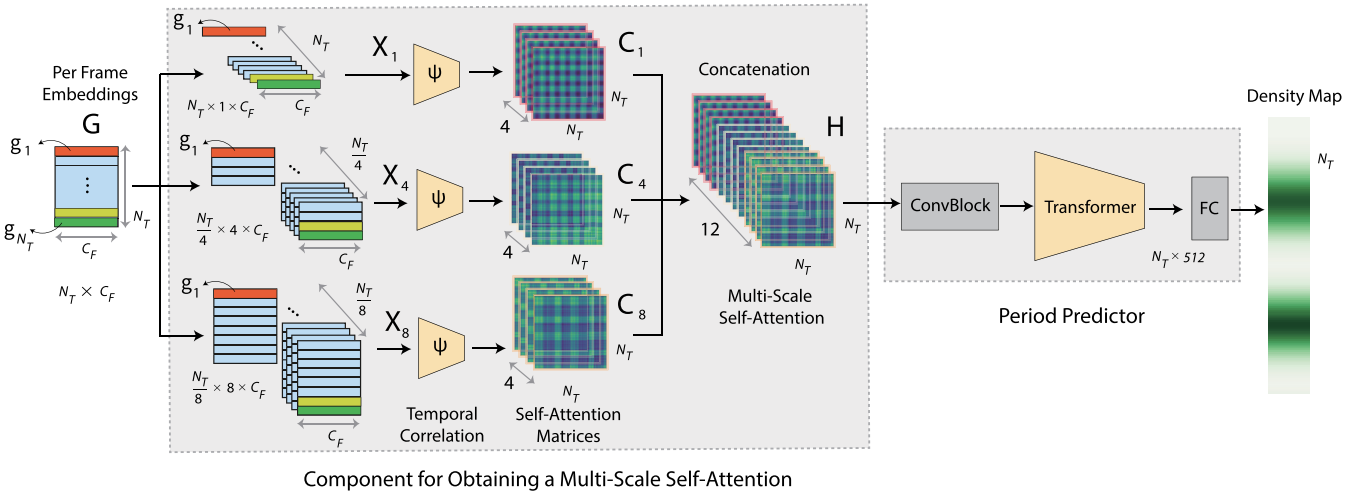


FIGURE 4. Multi-scale Self-attention matrix and period predictor.

where N_{T_k} is the number of video frames in the k-th video and $\|\cdot\|$ represents the L_2 norm (*i.e.*, Euclidean norm) of a vector. In Eq.(7), two N_{T_k}-dimensional vectors, $\hat{\mathbf{m}}^{(k)}$ and $\mathbf{m}^{(k)}$ are the estimated density map and the target density map of the k-th video, respectively.

In Eq.(5), $\mathcal{L}_{Count}$ is the mean absolute error (MAE) between the predicted count and the target count, which is defined as

$$\mathcal{L}_{Count} = \frac{1}{N_V}\sum_{k=1}^{N_V}\frac{|\hat{c}_k - c_k|}{c_k} \tag{8}$$

where $\hat{c}_k$ is the estimated count of repetitive action in the k-th video, and c_k is the corresponding target count. In Eq.(8), c_k and $\hat{c}_k$ can be computed from the k-th video using Eqs.(3) and (4), respectively.

Remark: In this paper, MAE represents the error metric $\mathcal{L}_{Count}$ defined in Eq.(8).

IV. DATA AUGMENTATION

As mentioned in Section II-C, there are very few publicly available datasets for RAC tasks in video, compared to datasets that support more general video understanding tasks. For this reason, data augmentation is required to improve the performance of RAC models. In [1], Dwibedi et al. presented a method to generate synthetic periodic videos from unlabeled videos, by inserting the repeating video frames into the original video and achieving camera motion augmentation. However, to perform camera motion augmentation, it is difficult to create a synthetic image from a given set of RGB images by changing camera parameters such as camera orientation and focal length. Compared to RGB images, skeletons are relatively easy to generate synthetically because they have fewer parameters and simple structure [23], [24], [25], [26].

In this section, we present a method to synthetically generate additional skeleton data by applying a linear transformation to the original skeleton data (see Fig. 5(c) and (d)). The procedure to augment the dataset is as follows:

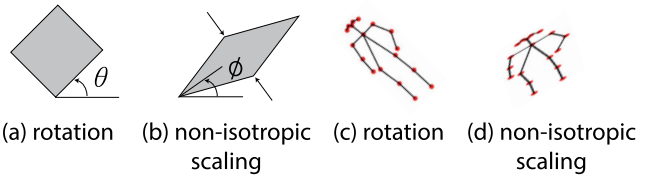


FIGURE 5. Data augmentation: We can generate different sets of 2D keypoints by applying linear transformations such as rotation and non-isotropic scaling to the original 2D skeletons.

- Repeat the following procedure until the number of augmented data reaches a preset value.
- We randomly sample three numbers $\theta, \phi, \eta \in \mathbb{R}$ within the following ranges: $\theta, \phi \sim$ Uniform$(-\pi/6, \pi/6)$ and $\eta \sim$ Uniform$(0.96, 1.04)$. Here Uniform(a, b) where $a, b \in \mathbb{R}$ represents a uniform distribution from a to b.
- We then generate N_T sets of Gaussian noises as follows: $\epsilon_i, \zeta_i \sim \mathcal{N}(0, 0.01)$ and $\xi_i \sim \mathcal{N}(0, 0.001)$, $(i = 1, \ldots, N_T)$. Recall that N_T is the number of consecutive video frames in a single video. Then, we add the Gaussian noises to θ, ϕ and η as follows: $\theta_i = \theta + \epsilon_i$, $\phi_i = \phi + \zeta_i$, and $\eta_i = \eta + \xi_i$.
- Using θ_i, ϕ_i and η_i, the i-th 2×2 non-singular matrix $\mathbf{A}_i$ can be created as follows:

$$\mathbf{A}_i = \mathbf{R}(\theta_i)\mathbf{R}(-\phi_i)\mathbf{D}(\eta_i)\mathbf{R}(\phi_i) \tag{9}$$

where $\mathbf{R}(\theta_i)$ and $\mathbf{R}(\phi_i)$ are rotations given by θ_j and ϕ_j, respectively (see Fig. 5(a) and (c)). The rotation matrix, $\mathbf{R}(\theta_i)$ is given by

$$\mathbf{R}(\theta_i) = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix}.$$

$\mathbf{D}(\eta_i)$ is a diagonal matrix of the form

$$\mathbf{D}(\eta_i) = \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix}. \tag{10}$$

which performs non-isotropic scaling (see Fig. 5(b) and (d)). In Eq.(10), we set $d_1 = 1$ and $d_2 = \eta_i$. Eq.(9) represents a singular value decomposition of $\mathbf{A}$ (For further details, refer to [27]): $\mathbf{A}_i = \mathbf{U}_i\mathbf{D}_i\mathbf{V_i}^T = (\mathbf{U_i}\mathbf{V_i}^T)(\mathbf{V}_i\mathbf{D}_i\mathbf{V_i}^T) = \mathbf{R}(\theta_i)(\mathbf{R}(-\phi_i)\mathbf{D}(\eta_i)\mathbf{R}(\phi_i))$ where $\mathbf{U}_i$ and $\mathbf{V}_i$ are orthogonal matrices.

- We randomly choose a video from Repcount-A [2] dataset.
- From the chosen video, we then extract a sequence of skeletons $\mathbf{S}_i, (i = 1, \ldots, N_T)$ using 2D pose estimator [15] as shown in Fig. 3. The j-th row of $\mathbf{S}_i$ represents the j-th 2D keypoints, $\mathbf{k}_{ji} \in \mathbb{R}^{1\times 2}$ in the i-th skeleton (see Eq.(1)).
- From the original skeleton data, $\mathbf{S}_i$, we can generate a new set of skeleton data, $\tilde{\mathbf{S}}_i, (i = 1, \ldots, N_T)$, as follows:

$$\tilde{\mathbf{S}}_i^{\,T} = \mathbf{A}_i\mathbf{S}_i^T. \tag{11}$$

The effectiveness of data augmentation is experimentally demonstrated in Table 8 of Section V-F3.

V. EXPERIMENTS

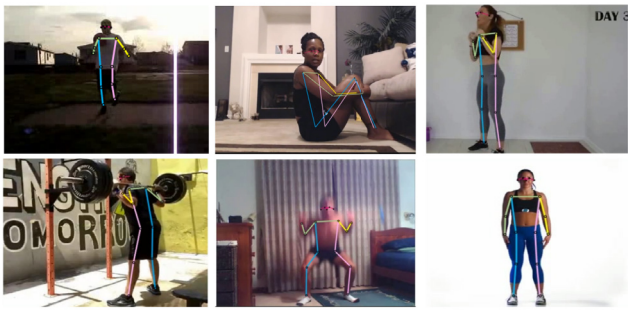
A. DATASETS

Among video datasets for RAC, the QUVA repetition dataset [21] and Countix [1] support more general and class-agnostic forms of RAC, such as periodic changes in illumination and the repetitive action of animal and human in videos. Compared to QUVA repetition dataset and Countix, RepCount-A [2] and UCFRep [16] focus more on repetitive *human* action in videos. Thus, we use RepCount-A and UCFRep to train and test our model that takes human skeletons as input.

Existing video datasets for RAC can be broadly classified into two categories: (i) coarse-grained datasets and (ii) fine-grained datasets. In the coarse-grained dataset (*e.g.*, the QUVA repetition dataset, Countix, and UCFRep), the number of repetitive actions in each video is annotated, but the start and end video frames of repetitive actions are not labeled. In contrast, the **fine-grained** dataset (*e.g.*, RepCount-A) contains labels for the start and end video frames of each repetitive action in videos.

1) REPCOUNT-A DATASET FOR TRAINING AND TESTING MODELS

Compared to coarse-grained dataset, fine-grained dataset can provide better interpretability to models [2]. For example, the total number of repetitions estimated by the model is correct, but the video frames at the start and end of the repetitive action in the video may be wrong [2]. In this case, the model may be incorrectly evaluated as performing well on a coarse-grained dataset, but in reality, the model did not perform well. Thus, a fine-grained dataset may be more suitable for model interpretation. For this reason, our model utilizes a fine-grained dataset such as RepCount-A dataset.



(a) RepCount-A Dataset

(b) UCFRep Dataset

FIGURE 6. Skeletons extracted from the videos in datasets using 2D pose estimator [15]: The images shown are from (a) the RepCount-A dataset [2] and (b) the UCFRep dataset [16].

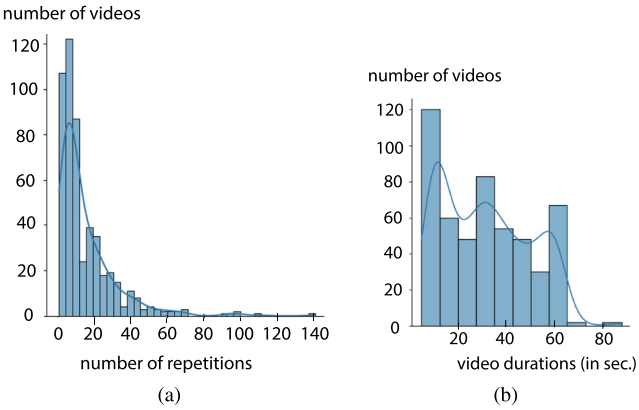


FIGURE 7. Statistics of the selected RepCount-A dataset [16] used in our experiments: histograms of (a) number of repetitions and (b) video durations.

Unfortunately, in the RepCount-A dataset [2] it is unclear who is actually performing the repetitive actions among multiple people. This problem was pointed out by the researchers who created the RepCount dataset in their paper [2]. Additionally, due to poor lighting in the video or other reasons, the 2D pose estimator may not be able to extract the skeleton. For the reasons mentioned above, we could not use the entire RepCount-A dataset for training and testing our model, but instead selected videos from the dataset where the 2D pose estimator recognized the human skeleton well.

We split the selected RepCount-A dataset [2] to use for model training, validation, and testing. See Table 2 in section V-D for a detailed explanation of how we split

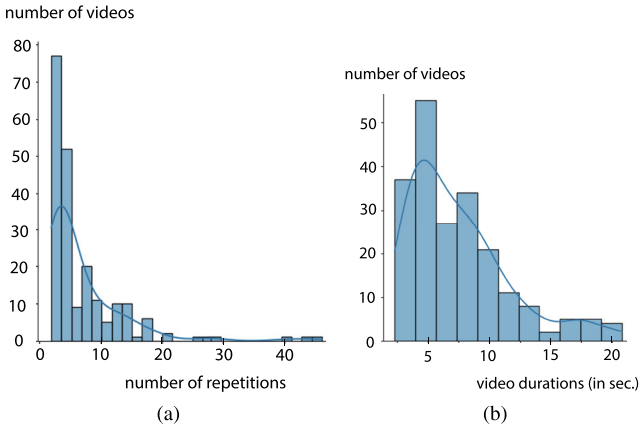


FIGURE 8. Statistics of the selected UCFRep dataset [2] used in our experiments: histograms of (a) number of repetitions and (b) video durations.

TABLE 1. Statistics of the datasets used in our experiments.

	RepCount-A [2]	UCFRep [16]
Numer of Videos	514	209
number of repetitions: mean / std.	15.226 / 16.489	6.832 / 6.657
number of repetitions: min. / max.	1 / 141	2 / 46
Duration: mean / std. (sec.)	31.621 / 18.001	7.47 / 4.091
Duration: min. / max. (sec.)	5 / 88	2.3 / 20.82

the training, validation, and test data. Fig. 6(a) shows the skeletons extracted from the RepCount-A [2] using the 2D pose estimator [15]. To ensure a fair comparison, both the existing models and our model were trained using the same dataset selected in this way.

2) UCFRep DATASET FOR TESTING MODELS

Unlike the RepCount-A [2], which is a fine-grained dataset, the UCFRep dataset [16] is a coarse-grained dataset where the start and end of the repetitive actions in videos are not labeled. So we should annotate the video frame numbers of the start and end of repetitive actions in videos from the UCFRep [16]. We first select videos from the UCFRep [16] where the 2D pose estimator is well recognized. The procedure mentioned above is almost the same as selecting videos from the RepCount-A dataset.

To add fine-grained annotations to the UCFRep dataset, we follow the labeling process for RepCount-A described in the paper [16] as follows: (i) Videos from the selected UCFRep are divided in half and assigned to two volunteers. (ii) Two volunteers annotate the start and end video frame numbers of repetitive actions in the video. (iii) Then, two volunteers perform cross-validation by comparing the annotations. If the count of repetitive actions labeled by two volunteers differs by more than one, they should revise their annotations.

3) DATASET STATISTICS

Figs. 7 and 8, and Table 1 show the statistics of the selected RepCount-A [2] and UCFRep [16] used in our

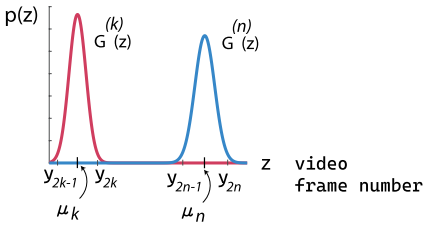


FIGURE 9. Generateion of multiple gaussian probability density functions (PDFs) from annotations that indicate the start and end frame numbers of repetitions in the video.

experiments. When looking at the statistics on the number of repetitions and duration in Table 1, we can see that the selected RepCount-A dataset has very similar statistics to the original RepCount-A. The UCFRep dataset we selected also has similar statistics to the original UCFRep in terms of the number of repetitions and duration (see Table 1).

B. GENERATING TARGET DENSITY MAPS FROM VIDEO DATASET

In this section, we briefly explain how the target density map, $\mathbf{m} = (m_1, \ldots, m_{N_T}) \in \mathbb{R}^{N_T}$, is generated from video dataset. Recall that N_T is the number of video frames. Through the process described below, we can generate the target density map from coarse-grained video datasets such as UCFRep [16]. For further details, see [2].

First, suppose that the start and end video frame numbers of repetitive action in a video are already manually annotated as follows: $\mathbf{y} = (y_1, y_2, \ldots, y_{2c})$ where c represents the total number of repetitive actions in the video. The goal of this paper is to estimate c value from the video frames. Then y_{2k-1} and $y_{2k}, (k = 1, \ldots, c)$ represent the start and end video frame numbers of the kth repetitive action in the video, respectively.

For $0 \le z \le N_T$, we can define a function $G^{(k)}(z)$ as

$$G^{(k)}(z) = \begin{cases} \mathcal{N}(z; \mu_k, \sigma_k), & \text{if } z \in [\mu_k - 3\sigma_k, \mu_k + 3\sigma_k] \\ 0, & \text{elsewhere} \end{cases} \tag{12}$$

where

$$\mathcal{N}(z; \mu_k, \sigma_k) = \frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{(z-\mu_k)^2}{2\sigma_k^2}}. \tag{13}$$

Here μ_k and σ_k are the mean and standard deviation of k-th repetitive action in the video frames so that $[y_{2k-1}, y_{2k}]$ corresponds to the 99% confidence interval of the Gaussian distribution. Since y_{2k-1} and y_{2k} are given by the manual annotation, we can obtain μ_k and σ_k by using the following equations: $y_{2k-1} = \mu_k - 3\sigma_k$ and $y_{2k} = \mu_k + 3\sigma_k$. For example, in Fig. 9, two functions $G^{(k)}(z)$ and $G^{(n)}(z)$ for the k-th and nth repetitive action show that the number of repetitive actions in this case is 2.

By using Eq.(12), the i-th component, m_i of the density map $\mathbf{m}$ can be calculated as

$$m_i = \int_{i-0.5}^{i+0.5} \sum_{k=1}^{c} G^{(k)}(z)dz, \quad i \in [1, 2, \ldots, N_T] \tag{14}$$

C. BENCHMARK AND EVALUATION METRICS

We evaluate our models on two video datasets: a subset of the RepCount-A dataset [2] and the UCFRep dataset [16]. Among the evaluation metrics for models predicting the number of repetitions in a video, MAE in Eq.(8) and OBO (Off-By-One) count are widely used [1], [2]. Let us assume that the model prediction is accurate if the predicted count is within 1 count of the ground truth value. Otherwise the prediction is considered inaccurate. From this perspective, the OBO count is defined as

$$\text{OBO} = \frac{1}{N_V} \sum_{k=1}^{N_V} \mathbf{I}_A(|\hat{c}_k - c_k| \leq 1) \tag{15}$$

where c_k and $\hat{c}_k$ can be obtained from the k-th video using Eqs.(3) and (4), respectively. Here N_V is the total number of videos. In Eq.(15), $\mathbf{I}_A(\cdot)$ represents an indicator function defined as

$$\mathbf{I}_A(a) = \begin{cases} 1 & \text{if } a \text{ is true} \\ 0, & \text{else.} \end{cases}$$

Models with low MAE in Eq.(8) and high OBO count in Eq.(15) are preferred.

D. IMPLEMENTATION DETAILS

We implemented the SSTRAC model using PyTorch. As shown in Fig. 1, our model consists of three components: (i) motion encoder, (ii) a component that obtains a multi-scale self-attention matrix, and (iii) period predictor. As a pre-processing step, we first extract 2D skeletal sequences from the video using an occlusion-robust 2D pose estimator [15] (see Figs. 3 and 6). In our implementation, the number of video frames, N_T in a video is equal to the number of skeleton sequences. We set $N_T = 256$. If the number of input video frames does not exceed 256, zero padding is applied to the skeleton sequence. In the motion encoder (see Fig. 2), the number of heads is 8, and the depth is $N = 3$ (*i.e.*, **Ours (Base)** model) or $N = 5$ (*i.e.*, **Ours (Large)** model).

Our model was trained for 200 epochs with a declining learning rate of 8×10^{-6} and optimized with the Adam optimizer using batch size 4 (*i.e.* 4 videos). For the training loss in Eq.(5), we set $\lambda_1 = 1$ and $\lambda_2 = 0.1$. We conducted experiments by running models on a single NVIDIA RTX 3080Ti GPU.

E. EVALUATION AND COMPARISON

In this section, we present experimental results comparing our model with other state-of-the-art models such as TransRAC [2] and PoseRAC [14] using subsets of RepCount-A [2] and UCFRep [16] (see Table 2). Here, PoseRAC requires additional work to annotate salient poses in the RepCount-A and UCFRep.

1) QUANTITATIVE COMPARISON

We first split the given RepCount-A [2] and UCFRep [16] (see Table 2). We trained our model and other state-of-the-art models using parts of RepCount-A (see Table 2). Hyperparameters for each model were set using the validation set of RepCount-A. We then performed comparative experiments using the test set, *i.e.*, a part of RepCount-A, and the UCFRep.

Table 3 summarizes the main experimental results, showing that our model (*i.e.*, Ours (Large) or Ours (Base)) outperforms existing state-of-the-art models. In Table 3, "w/o" and "Data Aug." mean "without" and "Data Augmentation", respectively. The data augmentation procedure for the training data is described in Section IV and V-F3. The expression "our dataset" in Table 3 refers to the dataset selected from RepCount-A and UCFRep described in Table 1. In Table 3, the expression "TransRAC (pre-trained)" means that a pre-trained TransRAC model was used in the experiments. Thus, all models in Table 3 except "TransRAC (pre-trained)" were trained using our dataset.

The superiority of our model to TransRAC and PoseRAC in Table 3 may result from the dual-stream spatio-temporal transformer in the motion encoder. Here, the motion encoder captures rich spatial and temporal information from the skeletal sequence and encodes them into per-frame embeddings.

As will be explained in the ablation study (see Section V-F2), abstracting the encoded per-frame embeddings using multi-scale self-attention in the model also contributes significantly to the model's superior performance over PoseRAC. This implies that PoseRAC in Table 3 may rely heavily on a small number of salient poses, which leads to relatively low performance.

Compared to TransRAC, our model is expected to require fewer parameters because it uses lower-dimensional skeletons as input. However, as shown in Fig. 10, Ours (Large) has more parameters than TransRAC due to the motion encoder. Note that the lightweight model (*i.e.*, Ours (Base)) is more accurate and has fewer parameters than TransRAC (see Table 3 and Fig. 10). If we ignore model lightness and only pursue lower MAE and larger OBO, Ours (Large) is better than Ours (Base).

TABLE 2. Dataset splitting: A subset of the RepCount-A dataset presented in Table 1 is randomly split into training, validation, and test sets containing 395, 53, and 66 videos, respectively. The subset of UCFRep in Table 1 is used only for model testing.

Number of videos	RepCount-A	UCFRep
Training	395	0
Validation	53	0
Testing	66	209

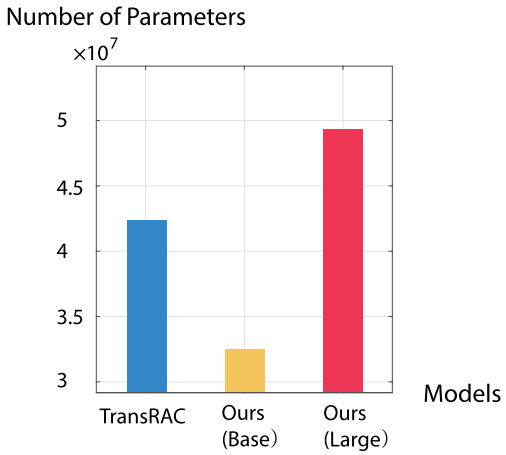


FIGURE 10. Comparison of number of parameters.

TABLE 3. Main experimental results (in case of no occlusion): We compare our model with the state-of-the-art models trained on selected datasets from RepCount-A and UCFRep, which are described in Table 1. As an exception, "TransRAC (pre-trained)" means a pre-trained TransRAC model.

	RepCount-A		UCFRep	
Models	MAE ↓	OBO ↑	MAE ↓	OBO ↑
Ours (Base) w/o Data Aug.	0.439	0.283	0.697	0.373
Ours (Large) w/o Data Aug.	0.414	0.303	0.687	0.378
Ours (Base) with Data Aug.	0.394	0.312	0.677	0.373
Ours (Large) with Data Aug.	**0.373**	**0.312**	**0.654**	**0.402**
TransRAC (our dataset) [2]	0.515	0.227	1.318	0.211
TransRAC (pre-trained) [2]	0.629	0.121	0.655	0.354
PoseRAC (our dataset) [14]	0.985	0.015	0.999	0.015

2) VISUAL COMPARISON

The ground truths for the number of repetitions in Fig. 11(a) and (b) are 3 and 2, respectively. In Fig. 11, dark areas on the target density map indicate areas where repetitive actions occur. Fig. 11(a) and (b) visually show that the proposed model (*i.e.* Ours (Base)) outperforms the existing state-of-the-art model, transRAC, in terms of density map prediction.

3) HANDLING CASES WHERE OBJECTS OCCLUDE A PERSON IN THE VIDEO

When predicting the number of repetitive actions in a video, the target is usually a person. However, it is common for people to be cropped or occluded by other objects (*e.g.*, pets or furniture) in the video. To the best of our knowledge, existing models for RAC do not address this occlusion problem in videos.

In existing models (*e.g.*, TransRAC [2]) that directly extract RGB features from images, the network structure can become complex if such occlusion is taken into account. In contrast, our model can solve this occlusion problem independently of the network structure. Our model addresses the occlusion problem by reconstructing the defective 2D skeleton using a pre-trained occlusion-robust 2D pose estimator [15].

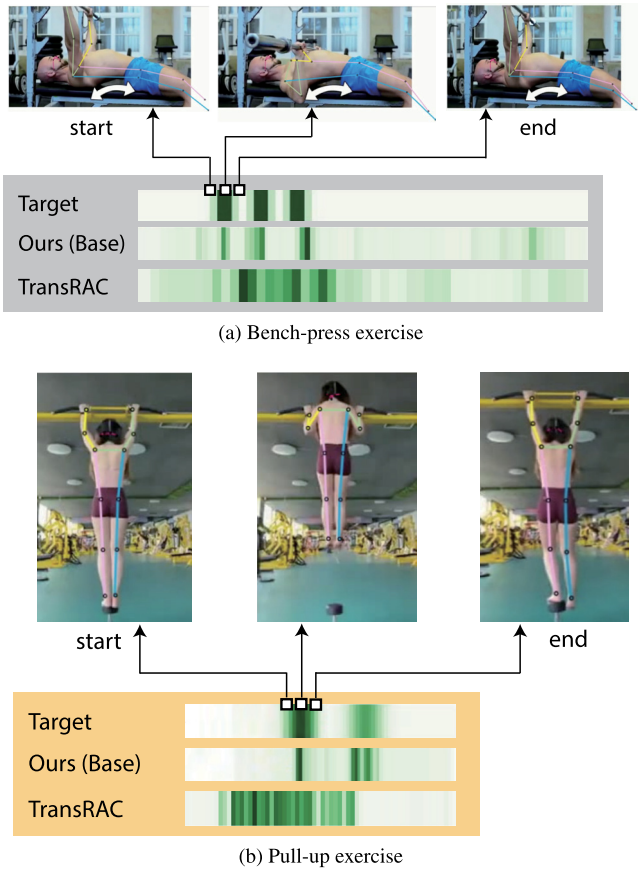


FIGURE 11. Visual comparison (in case of no occlusion).

To simulate the occlusion problem, we generate synthetic objects that occlude a person in the video of RepCount-A. In our experiment, the synthetic object is a basketball, which initially appears on the right side of the video. The ball moves from right to left based on a random function with a maximum pixel range of [-2,1] per video frame. As a result, the ball tends to move leftward across the video, with its speed randomly determined. The reason for creating a synthetic object from a video is that we can know the ground-truth of the number of repetitions through simulation.

From Table 4 and Figure 12, we can see that our model outperforms TransRAC in the presence of synthetic occlusion.

TABLE 4. Experimental results of models using the selected RepCount-A dataset described in Table 2 (in case of occlusion): In this experiment, a synthetically generated basketball moves slowly from right to left in the video (see Fig. 12).

	RepCount-A		UCFRep	
Models	MAE ↓	OBO ↑	MAE ↓	OBO ↑
Ours (Base)	0.444	0.272	0.698	0.340
Ours (Large)	**0.418**	**0.288**	**0.663**	**0.397**
TransRAC (our dataset) [2]	0.611	0.227	1.302	0.215
TransRAC (pre-trained) [2]	0.643	0.212	0.673	0.368

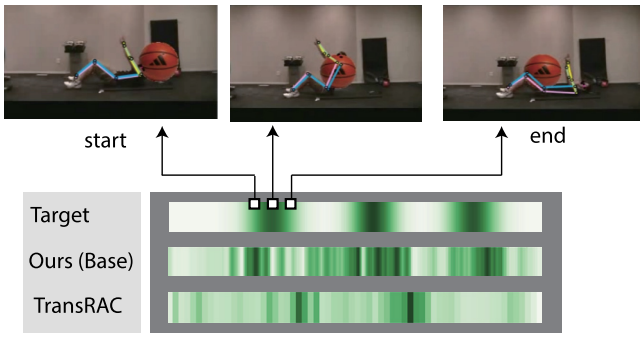


FIGURE 12. Visual comparison (in case of occlusion).

4) RUNTIME PERFORMANCE

In this section, we present the runtime performance of our model by measuring the inference speed. To measure inference speed, we ran the model on a desktop computer equipped with an Intel i9-11900KF CPU and an RTX 3080Ti GPU. Table 5 shows that the motion encoder in our model consumes the most computational time during the inference process.

TABLE 5. Average computation time for each component of our model (Ours (Base)): Units are milliseconds (msec).

Motion Encoder	Multi-Scale Self Attention	Period Predictor	Total
8.91	1.93	2.19	13.03

F. ABLATION STUDY

As an ablation study, we empirically evaluate the impact of each component of the SSTRAC model (see Figs. 1, 2, and 4) on model performance using the selected RepCount-A dataset described in Table 1. In this ablation study, the depth of the motion encoder is set to $N = 5$ (*i.e.*, Ours (Large)). We then unveil the effectiveness and rationality of the proposed model.

1) DUAL-STREAM ARCHITECTURE OF MOTION ENCODER

The motion encoder in Fig. 2 has a dual-stream architecture that fuses spatio-temporal stream and temporal-spatial stream. We performed ablation studies by removing the spatio-temporal stream or the temporal-spatial stream (see Table 6). Here, the spatio-temporal stream represents a single stream in which the spatial MHSA comes first, followed by the temporal MHSA. Similarly, the temporal-spatial stream has the temporal MHSA followed by the spatial MHSA. Table 6 shows that the model with dual-stream architecture performs best.

2) MULTI-SCALES FOR CALCULATING SELF ATTENTION MATRICES

The multi-scale self-attention matrix shown in Fig. 4 contains abstracted features for repetitive actions with different period lengths in the skeletal sequence. We conducted experiments

TABLE 6. Ablation study on dual-stream architecture for motion encoder.

Single or Dual Stream	MAE ↓	OBO ↑
Single-Stream (Only Spatio-Temporal)	0.804	0.091
Single-Stream (Only Temporal-Spatial)	0.696	0.075
Dual-Stream	**0.414**	**0.303**

changing combinations of three different scales, *i.e.*, scale 1, 4, and 8 (see Fig. 4 and Eq.(2)). Table 7 shows that the model using all three scales (*i.e.*, scale 1, 4, and 8) performs best. Having more diverse scales can improve the performance of models, but may come at the expense of increased computational complexity.

TABLE 7. Ablation study on multi-scales for obtaining self attention matrices.

Combination of scales	MAE ↓	OBO ↑
Scale 1	0.635	0.182
Scale 8	0.656	0.152
Scale 1 and 4	0.599	0.227
Scale 1 and 8	0.573	0.212
Scale 1, 4, and 8	**0.414**	**0.303**

3) DATA AUGMENTATION

This section explains how to augment the training data in RepCount-A. As shown in Table 2, the number of videos used for training models in the RepCount-A data is 395. First, we randomly sample 105 of the 395 videos used to train the model in the RepCount-A dataset. We extract skeleton sequences from the 105 videos using the 2D pose estimator [15], and then apply an affine transformation to them for generating a new set of skeleton sequences (see Section IV). Thus, the augmented dataset consists of skeleton sequences extracted from the original 395 videos and newly generated skeleton sequences from 105 sampled videos.

Table 8 demonstrates that data augmentation explained in Section IV is effective. Table 8 shows only the experimental results corresponding to Ours (Base) and Ours (Large) in Table 3.

TABLE 8. Ablation study on data augmentation.

	RepCount-A		UCFRep	
Models	MAE ↓	OBO ↑	MAE ↓	OBO ↑
Ours (Base) w/o Data Aug.	0.439	0.283	0.697	0.373
Ours (Large) w/o Data Aug.	0.414	0.303	0.687	0.378
Ours (Base) with Data Aug.	0.394	0.312	0.677	0.373
Ours (Large) with Data Aug.	**0.373**	**0.312**	**0.654**	**0.402**

VI. CONCLUSION AND LIMITATIONS

The goal of this paper is to develop a RAC model that is robust to changes in human size and occlusion of body parts in videos. To achieve this goal, we presented a novel SSTRAC model consisting of motion encoder, a component for obtaining a multi-scale self-attention matrix,

and period predictor. The proposed model takes a sequence of skeletons as input and can encode rich spatial and temporal information through the dual-stream spatio-temporal transformer of the motion encoder. Per frame embeddings encoded by the motion encoder are then abstracted into a multi-scale self-attention matrix shown to capture human repetitive actions with different period lengths. In the final step, the period predictor estimates a density map, which provides the number of repetitive actions in each video. Extensive experiments have shown that our model outperforms other state-of-the-art models in terms of robustness to the change of human body size and the occlusion of human body parts in videos.

Limitations: The performance of the proposed model is affected by the 2D pose estimator used in the preprocessing step. The 2D pose estimator [15] used in this paper is robust to occlusion, and the dataset contains videos with little occlusion or blur. However, its performance may be degraded under severe occlusion or motion blur.

Most 2D pose estimators have limitations in that they are not robust to camera viewpoint changes and camera distortion. To address these issues when using the 2D pose estimator for RAC, we augmented our dataset by applying rotation and non-isotropic scaling to the training data, which consists of 2D skeletons. Applying rotation and non-isotropic scaling to the 2D skeletons allows us to approximate changes in camera viewpoint, camera distortion, and 2D skeleton distortion (see Fig. 5(c) and (d)).

REFERENCES

[1] D. Dwibedi, Y. Aytar, J. Tompson, P. Sermanet, and A. Zisserman, "Counting out time: Class agnostic video repetition counting in the wild," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10384–10393.

[2] H. Hu, S. Dong, Y. Zhao, D. Lian, Z. Li, and S. Gao, "TransRAC: Encoding multi-scale temporal correlation with transformers for repetitive action counting," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 18991–19000.

[3] K. Li, X. Peng, D. Guo, X. Yang, and M. Wang, "Repetitive action counting with hybrid temporal relation modeling," *IEEE Trans. Multimedia*, vol. 27, pp. 3844–3855, 2025.

[4] Z. Zhang, K. Li, S. Tang, Y. Wei, F. Wang, J. Zhou, and D. Guo, "Temporal boundary awareness network for repetitive action counting," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 21, no. 4, pp. 1–22, Apr. 2025.

[5] Z. Yao, X. Cheng, Z. Huang, and L. Li, "CountLLM: Towards generalizable repetitive action counting via large language model," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2025, pp. 19143–19153.

[6] W. Zhu, X. Ma, Z. Liu, L. Liu, W. Wu, and Y. Wang, "MotionBERT: A unified perspective on learning human motion representations," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 15039–15053.

[7] H. Huang, S. Gou, R. Li, and X. Gao, "Joint-wise temporal self-similarity periodic selection network for repetitive fitness action counting," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 34, no. 10, pp. 9808–9821, Oct. 2024.

[8] C. Li, M. Shao, Q. Yang, and S. Xia, "High-precision skeleton-based human repetitive action counting," *IET Comput. Vis.*, vol. 17, no. 6, pp. 700–709, Sep. 2023.

[9] C. Wang and J. Yan, "A comprehensive survey of RGB-based and skeleton-based human action recognition," *IEEE Access*, vol. 11, pp. 53880–53898, 2023.

[10] W. Xin, R. Liu, Y. Liu, Y. Chen, W. Yu, and Q. Miao, "Transformer for skeleton-based action recognition: A review of recent advances," *Neurocomputing*, vol. 537, pp. 164–186, Jun. 2023.

[11] Y.-F. Song, Z. Zhang, C. Shan, and L. Wang, "Constructing stronger and faster baselines for skeleton-based action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 2, pp. 1474–1488, Feb. 2023.

[12] Y. Du, Y. Fu, and L. Wang, "Representation learning of temporal dynamics for skeleton-based action recognition," *IEEE Trans. Image Process.*, vol. 25, no. 7, pp. 3010–3022, Jul. 2016.

[13] H.-G. Chi, M. H. Ha, S. Chi, S. W. Lee, Q. Huang, and K. Ramani, "InfoGCN: Representation learning for human skeleton-based action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 20154–20164.

[14] Z. Yao, X. Cheng, and Y. Zou, "PoseRAC: Pose saliency transformer for repetitive action counting," 2023, *arXiv:2303.08450*.

[15] Z. Geng, C. Wang, Y. Wei, Z. Liu, H. Li, and H. Hu, "Human pose as compositional tokens," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 660–671.

[16] H. Zhang, X. Xu, G. Han, and S. He, "Context-aware and scale-insensitive temporal repetition counting," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 667–675.

[17] X. Li and H. Xu, "Repetitive action counting with motion feature learning," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2024, pp. 6485–6494.

[18] Y. Kong and Y. Fu, "Human action recognition and prediction: A survey," *Int. J. Comput. Vis.*, vol. 130, no. 5, pp. 1366–1401, May 2022.

[19] M. Karim, S. Khalid, A. Aleryani, J. Khan, I. Ullah, and Z. Ali, "Human action recognition systems: A review of the trends and state-of-the-art," *IEEE Access*, vol. 12, pp. 36372–36390, 2024.

[20] A. Stergiou and R. Poppe, "About time: Advances, challenges, and outlooks of action understanding," *Int. J. Comput. Vis.*, vol. 133, no. 9, pp. 6251–6315, Sep. 2025.

[21] T. F. H. Runia, C. G. M. Snoek, and A. W. M. Smeulders, "Real-world repetition estimation by div, grad and curl," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9009–9017.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NeurIPS*, vol. 30, 2017, pp. 5998–6008.

[23] C. Xin, S. Kim, Y. Cho, and K. S. Park, "Enhancing human action recognition with 3D skeleton data: A comprehensive study of deep learning and data augmentation," *Electronics*, vol. 13, no. 4, p. 747, Feb. 2024.

[24] K. Fukushi, Y. Nozaki, K. Nishihara, and K. Nakahara, "Few-shot generative model for skeleton-based human action synthesis using cross-domain adversarial learning," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2024, pp. 3934–3943.

[25] F. Meng, H. Liu, Y. Liang, J. Tu, and M. Liu, "Sample fusion network: An end-to-end data augmentation network for skeleton-based human action recognition," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5281–5295, Nov. 2019.

[26] S.-S. Park, H.-J. Kwon, J.-W. Baek, and K. Chung, "Dimensional expansion and time-series data augmentation policy for skeleton-based pose estimation," *IEEE Access*, vol. 10, pp. 112261–112272, 2022.

[27] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed., Cambridge, U.K.: Cambridge Univ. Press, 2003.

JUNGJUN LIM received the B.S. degree in electrical engineering from Korea University, Seoul, in 2025. From 2023 to 2024 (between semesters), he was a Student Researcher with the Intelligence and Interaction Research Center, Korea Institute of Science and Technology (KIST). His research interests include computer vision, deep learning, and large language models with multimodal learnings.

DONGHOON KANG (Member, IEEE) received the B.S. and M.S. degrees in mechanical engineering from Pohang University of Science and Technology (POSTECH), Pohang, South Korea, in 1997 and 1999, respectively, and the Ph.D. degree in mechanical and aerospace engineering from Seoul National University, Seoul, South Korea, in 2018.

He has been with Korea Institute of Science and Technology (KIST), since 2000, and is currently a Senior Researcher. He has been an Adjunct Professor with the Division of AI-Robotics, KIST School, University of Science and Technology (UST), since 2020. His research interests include latent variable optimization in robotics and machine learning, and video understanding using deep learning.

Dr. Kang received the Gold Medal from IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT for his outstanding performance as one of the most productive reviewers over the past seven years (2014–2020). He was also selected as one of the Outstanding Reviewers for IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT (2017–2020). From 2021 to 2024, he was an Associate Editor of IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, specializing in robotics and artificial intelligence.

KANGHYUN RYU (Member, IEEE) received the B.S. and Ph.D. degrees in electrical engineering from Yonsei University, Seoul, South Korea, in 2015 and 2020, respectively.

From 2020 to 2022, he was a Postdoctoral Researcher with the Department of Radiology, Stanford University. Since 2022, he has been a Senior Researcher was Korea Institute of Science and Technology (KIST). In 2025, he joined the AI-Robotics Division, KIST School, University of Science and Technology (UST), as an Assistant Professor. He has authored more than 20 SCIE-indexed journal articles. His research interests include developing innovative technologies to enhance human health and quality of life by integrating healthcare imaging modalities, such as MRI and CT with the IoT-based monitoring systems, including CCTV cameras and sensors.

JE HYEONG HONG (Member, IEEE) received the B.A. and M.Eng. degrees in engineering (electrical and information sciences) and the Ph.D. degree in engineering (computer vision) from the University of Cambridge, U.K., in 2011 and 2018, respectively.

After collaborating with Microsoft and Toshiba Research Europe, he was in alternative military service in South Korea as a Postdoctoral Researcher with Korea Institute of Science and Technology (KIST), from 2018 to 2021. He has been an Assistant Professor with the Department of Electronic Engineering, Hanyang University, South Korea, since 2021. He has published several papers at prestigious conferences, such as CVPR, ECCV, ICCV, and ICASSP. His main research interests include computer vision, machine learning, and optimization.