Encoding Multi-scale Temporal Correlation with Transformers for Repetitive Action Counting

Supplementary Material

A. Extra experiments

A.1. Density map

Gaussianization. Assuming we have the label as $Y = [y1, y2, \ldots, y_n]$. A pair of y_i and y_j represents the frames at the beginning and end of a repetition action where $j = i+1$. To calculate the Gaussian function [2] with 99% confidence interval shown below, we need to figure out the mean μ and the variance σ. Because of 99% confidence interval meaning $\mu \pm 3\sigma$, we could get μ and σ by $y_{i,j} = \mu \pm 3\sigma$, where i and j is the pair of the frames.

Therefore, through the Gaussian function $g_\sigma(x)$, we can get the probability density distribution $G_\sigma(y)$ from y_i to y_j. Then d_k can take the integral by Eq. (1). At last, we could get the predict results of density map $D = [d_1, d_2, \ldots, d_n]$.

$$d_k = \int_{y_k-0.5}^{y_k+0.5} G_\sigma(y)dy, \quad k \in [i, j] \tag{1}$$

To compare different generate methods of density map, We adjust the mean μ of the Gaussian function to the beginning frame of one period or the ending and retrain the model which has a merging density map predictor. Then we obtain the output of different positions by adjusting the weight of predict density maps.

	RepCount A	
Generate density map	MAE↓	OBO ↑
Begin	0.5295	0.2052
Mid	0.4936	0.2052
End	0.5192	0.192
Merge	0.5142	0.2009

Table 1. **The density maps with different mean μ of the Gaussian function G.** *Begin* is means the density map generated with G, where the μ is the beginning frame. Similarly, *End* represents the μ is the ending frame. *Mid* is as same as our model **TranRAC**.

The result as the Tab. 1 shown, Merging density maps does not give the models better performance. Because the amount of video frames is 64, moving μ to begin or end will lose the information of the first period or the last. The density map generated by the mean in the mid-frame has the best effort.

A.2. Sample rate

We conducted the experiment to verify the impact of adding the number of video frames. Due to our model based on the density map, we use a one-dimensional spatial distribution to represent the distribution of periods in time.

	RepCount A	
Method	MAE↓	OBO ↑
Ours(single)-64	0.6595	0.185
Ours(single)-128	0.6191	0.191

Table 2. Experiment results of model with different sample rate when trained on train set of *RepCount part-A*. 64 and 128 indicates different frame sample number from initial videos.

Experimental results show that increasing video frames can improve the performance of density maps to a certain extent (see Tab. 2). A better result in terms of MAE error is achieved when we select 128 frames of a video.

A.3. Scale ablation

We verify the effect of different scales by building tree pipelines, where the input of Encoder ϕ with distinct length video subsequences. As shown in the one to three rows of Tab. 4, because the different temporal scales of video subsequence extract information in their scale, they have different performances of repetition counting. Concatenating the multi-scale video sequences contributes to capturing different period length actions and brings the model greater robustness.

A.4. Receptive field

Usually, a more large video subsequence of scale has a more substantial receptive field. And considering the more large sample rate will extract more video information, we compared the different video subsequences of scale at different sample rates.

As Tab. 5 indicated, when the sample rate is the little one, 64 frames extracted from one original video, single-frame is as similar as 8-frames. But increasing the sample rate to 128, The performance of 8-frames is far better than that of single-frames.

We believe that there is an optimal scale of video subsequence for the same dataset under the same sampling rate. Due to the operation of sampling video to the fixed number of frames, the duration of per repetitive action will be shorter with the decrease of the sample rate. Large-scale video sequences will lose their advantages when the sample rate is 64 frames per video because of the shorter duration. But when the sampling rate increases to 128, there

Division	Regular setting		Open-set setting	
	Num. of videos	Total frames	Num. of videos	Total frames
Train	758	637,545	655	560,402
val	131	109,854	130	77,103
test	152	129,993	256	239,887

Table 3. Regular setting and Open-setting of *RepCount partA*

	RepCount A	
Method	MAE↓	OBO ↑
Scale-1	0.6595	0.1854
Scale-4	0.5434	0.2649
Scale-8	0.6657	0.192
Ours	0.4431	0.2913

Table 4. **The experimental results of pipelines with different scales.** *Scale-i*, where $i \in \{1, 4, 8\}$, represents the temporal length of video subsequence, which is the input of Encoder ϕ. The *Ours*, means cancatenating three scales video subsequence together, have the lowest MAE. We build all the above models by extracting 64 frames from the original video.

	RepCount A		
Samle rates	Scales	MAE↓	OBO ↑
64	Scale-1	0.6595	0.185
	Scale-8	0.6657	0.192
128	Scale-1	0.6191	0.191
	Scale-8	0.4926	0.2302

Table 5. **Performance of different scales at different sample rates.** The first column, *sample rates*, indicates different frame sample number from initial videos. *Scale-i*, where $i \in \{1, 8\}$, represents the temporal length of video subsquence.

is more difference reflected between different video subsequence scales. Although single-frame has improved the MAE and OBO, the progress of 8-frames is more excellent.

A.5. Compare to action segmentation

We elaborate the definitions and differences between action segmentation and repetitive action counting. Given an input video, action segmentation is to segment the temporal bound for different types of actions but repetitive action counting aims to count the number of repetitive action [1, 4]. Two main differences are as follows: i) for action segmentation, the same action continuously repeating many times will be segmented into a single temporal bound. Thus, it is difficult to handle videos with high-frequency repetitive actions. However, the variation in the frequency of repetitive action is huge, *e.g.*, the min/max cycle length is 0.1/10.96 in our dataset, which degenerates action segmentation method; ii) action segmentation can only address predefined action types and cannot handle open-set setting, where action types in the test set that do not exist in the training set. However, repetitive counting is to record repeated actions regardless of the action category.

To verify the above differences, we further perform experiments in Tab. 6. Under both settings of Tab. 3, our method achieves better performance compared to action segmentation [3]. Therefore, our task is not a trivial case of action segmentation.

Method	regular setting		Open-set setting	
	MAE↓	OBO↑	MAE↓	OBO↑
Huang et al. [3]	0.5267	0.1589	1.0000	0.0000
Ours	0.4431	0.2913	0.6249	0.2040

Table 6. Performance of different methods on two settings of *RepCount partA*.

B. Dataset description

B.1. Data duration

The duration in Table 1 means video length, not the cycle length of each action, which shows that our dataset contains longer videos. In addition, not only the short action but the diversity of actions makes the task harder and more useful. The min/max cycle length between Ours and UCF526 [4] is (0.1/10.96 vs 0.12/6.76), which shows our dataset is more challenging.

B.2. Open-set setting

We add an open-set setting to demonstrate the better ability of our method when dealing with unseen action types in the training set. Therefore, we re-split the *RepCount partA* into a new train/val and test subset. For regular settings, videos are divided randomly. For Open-set setting, the action types in train/val/test are disjoint, where the actions in the test set do not appear in the training set. More details show in Tab. 3.

References

[1] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. Counting out time: Class agnostic video repetition counting in the wild. In *IEEE/CVF*

Conference on Computer Vision and Pattern Recognition (CVPR), June 2020. 2

[2] Hongwei Guo. A simple algorithm for fitting a gaussian function [dsp tips and tricks]. *IEEE Signal Processing Magazine*, 28(5):134–137, 2011. 1

[3] Yifei Huang, Yusuke Sugano, and Yoichi Sato. Improving action segmentation via graph-based temporal reasoning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14024–14034, 2020. 2

[4] Huaidong Zhang, Xuemiao Xu, Guoqiang Han, and Shengfeng He. Context-aware and scale-insensitive temporal repetition counting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2