ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-7605-5/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICASSP39728.2021.9413650

FINE-GRAINED POSE TEMPORAL MEMORY MODULE FOR VIDEO POSE ESTIMATION AND TRACKING

Chaoyi Wang[*], *Yang Hua*[†], *Tao Song*[*], *Zhengui Xue*[*], *Ruhui Ma*[*], *Neil Robertson*[†], *Haibing Guan*[*]

[*] Shanghai Jiao Tong University, China
[†] Queen's University Belfast, UK

ABSTRACT

The task of video pose estimation and tracking has been largely improved with the development of image pose estimation recently. However, there are still many challenging cases, such as body part occlusion, fast body motion, camera zooming, and complex background. Most existing methods generally use the temporal information to get more precise human bounding boxes or just use it in the tracking stage, but they fail to improve the accuracy of pose estimation tasks. To better solve these problems and utilize the temporal information efficiently and effectively, we present a novel structure, called pose temporal memory module, which is flexible to be transferred into top-down pose estimation frameworks. The temporal information stored in the pose temporal memory is aggregated into the current frame feature in our proposed module. We also transfer compositional de-attention (CoDA) to solve the unique keypoint occlusion problem in this task and propose a novel keypoint feature replacement to recover the extreme error detection under fine-grained keypoint-level guidance. To verify the generality and effectiveness of our proposed method, we integrate our module into two widely used pose estimation frameworks and obtain notable improvement on the PoseTrack dataset with only a few extra computing resources.

Index Terms— video pose estimation and tracking, keypoint occlusion

1. INTRODUCTION

In recent years, human pose estimation has become one of the fundamental research topics in computer vision and has made fast progress with the development of deep learning. With the publicly available video dataset PoseTrack [1] being introduced, human pose estimation and articulated tracking in video has become a new challenging task, which has more potential applications in real-world scenarios, such as action analysis, motion capture, and human interaction understanding.

Compared with the tasks using image data, the video-based pose estimation task is prone to problems, including motion blur because of fast body motion or camera zooming, body part occlusion for people interaction, and strange poses with action changing. These common problems cannot be solved properly by applying the image-based approaches [2, 3] directly on video data, since they barely utilize the temporal information in the pose estimation process.

To address these issues, many video-based human pose estimation approaches have been proposed. For example, ProTracker [4] uses a 3D convolution encoding temporal context within a sliding temporal window on the video sequence. However, this method is ineffective and has high computational complexity. LightTrack [5] applies a siamese graph convolution network for pose matching in tracking but it ignores the precision improvement of the pose estimation. FlowTrack [6] propagates the optical flow of the joints in previous frames, which inputs a more precise bounding box into the single person pose estimation network, but it still uses the image-based method after cropping the human instance from the enhanced box, which belongs to the "box-level" method.

Inspired by the "feature-level" method which utilizes the temporal information to get more precise feature maps instead of bounding boxes in the video object detection field [7], we propose a novel fine-grained pose temporal memory module, which is a lightweight addon and can be easily integrated into other top-down frameworksDifferent from other memory (a.k.a., hidden state) in a Recurrent Neural Network (RNN) [8], the size of the pose temporal memory is variable and has the potential to carry large-scale temporal information.

It is worth noting that the video pose estimation and tracking task has fine-grained targets (i.e., keypoints) comparing with the normal object in the video object detection task. The most notable difference between these two approaches is how the occlusion is handled. The occlusion in video object detection is only partial occlusion, since we do not need to detect the fully occluded objects. In contrast, the video pose estimation and tracking task targets at localizing all keypoints, even they are fully occluded. The vanilla attention [9], used for video object detection, can only re-weight the input features to enhance the similar areas between frames but suppress the

Tao Song is the corresponding author.

This work was supported in part by National NSF of China (NO. 61872234, 61732010, 61525204) and Shanghai Key Laboratory of Scalable Computing and Systems.

dissimilar areas including the occlusion, which is not suitable to solve the unique keypoint occlusion problem in the video pose estimation and tracking task. To address this problem, inspired by the Compositional De-Attention (CoDA) [10] from the natural language processing field, we additionally learn a dissimilarity matrix of two features at the pixel level in the memory module and get better performance than the vanilla attention when keypoints are occluded.

Furthermore, to manipulate the proposed dynamic fine-grained pose temporal memory effectively and efficiently, a set of *read* and *write* operations is designed. The *read* operation focuses on propagating variable sized memory accurately to the current frame while it also balances the present feature signal strength in the temporal enhanced feature map. The *write* operation selectively stores new features based on the quality of detection result. To deal with the extreme keypoint occlusion cases, we also propose a novel method, called keypoint feature replacement, in *write* operation. It refines the "feature-level" method in the video object detection field to "keypoint-level" method in the video pose estimation and tracking task. As a result, the temporal enhanced feature is partially replaced by the high-quality keypoint feature region in extreme occlusion conditions.

2. METHOD

2.1. Framework Overview

Our proposed pose temporal memory module is a general add-on to utilize the temporal information effectively and efficiently, which can be integrated into top-down frameworks. Here we use the SimpleBaseline [6] as our backbone detector to explain the pose temporal memory module, as shown in Figure 1. Given a video sequence, we first use the human detector to crop the single person from each frame in an online manner as our framework input. Then the original feature map F is generated from the feature extractor. Through the *read* operation, the pose temporal memory which is related to the current detecting instance propagates the stored feature maps generated in previous frames detection to the module with original feature map F, and a temporal enhanced feature map $\tilde{F}$ is produced. Then $\tilde{F}$ is used to produce the final heatmap H under the upsampling operation. Based on the quality of the heatmap H, the temporal enhanced feature map $\tilde{F}$ is refined using keypoint feature replacement under keypoint-level guidance and stored into the corresponding memory by the *write* operation.

2.2. Fine-Grained Pose Temporal Memory

The pose temporal memory is the core of our proposed module, which is a place to store the features from the same person instance in a video. During the pose estimation and tracking process, every people has a unique pose temporal memory. The size of the pose temporal memory grows when new features are written into it, and the saved features are permanently stored unless the person in the video disappears or the size is out of capacity. To clearly express the mechanism of the pose temporal memory module, we would focus on describing the two novel operations *read* and *write* in detail from the perspective of the pose temporal memory.

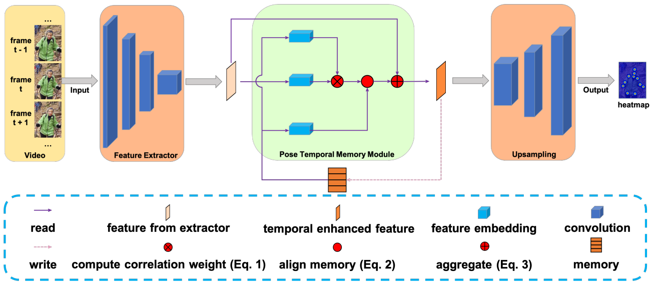

Fig. 1: Pose temporal memory module based on SimpleBaseline backbone. *Best viewed in color.*

***Read* operation** contains two stages. In the first stage, it needs to find the corresponding pose temporal memory from the previous detection result. In this paper, we simply use the IoU to find the human instance in the previous frame associated with the current detecting human instance. More specifically, we first calculate the bounding box IoU between the current detecting human instance with all the human instances in the previous frame, then we select the maximum one and propagate the pose temporal memory to the current detecting human instance. If the maximum IoU is lower than a threshold, i.e. a new person appears in the video, new pose temporal memory space will be applied for the new person.

The second stage involves three steps, including computing correlation weight by Eq. (1), aligning memory using Eq. (2) and aggregating memory with the input feature map by Eq. (3). Let $\boldsymbol{m} \in \mathbb{R}^{l \times c}$ be the pose temporal memory matrix, with the transpose of the i^{th} row denoted as $\boldsymbol{m_i}$. Let $\boldsymbol{x} \in \mathbb{R}^{n \times c}$ be the corresponding input feature matrix reshaped from F, with the transpose of the j^{th} row denoted as $\boldsymbol{x_j}$. Notation l represents a variable value and it equals to the multiplication of the number of features stored in the pose temporal memory and the feature shape n. The feature channel and embedding channel are denoted as c and $\overline{c}$, respectively. The degree of crowdedness of a frame is used here to measure the probability of the keypoint occlusion. First, $\boldsymbol{m}$ and $\boldsymbol{x}$ are transformed into two feature spaces K, Q to compute the correlation weights $w = \{w_{i,j}\}$ by CoDA or vanilla attention according to

$$w_{i,j} = \begin{cases} tanh(s_{ij}) \odot (2 * sigmoid(d_{ij})) & \text{crowded} \\ \exp(s_{ij}) / \sum_{i=1}^{l} \exp(s_{ij}) & \text{uncrowded}, \end{cases} \tag{1}$$

where s_{ij} and d_{ij} evaluate the similarity and dissimilarity between $\boldsymbol{m_i}$ and $\boldsymbol{x_j}$, respectively. They are calculated by $s_{ij} = (W_K \boldsymbol{m_i})^T (W_Q \boldsymbol{x_j})$ and $d_{ij} = -\left\| W_K \boldsymbol{m_i} - W_Q \boldsymbol{x_j} \right\|_{l_1}$ with

$W_K \in \mathbb{R}^{\bar{c}\times c}$ and $W_Q \in \mathbb{R}^{\bar{c}\times c}$ being learnable embedding weights. Notation $\odot$ is the element-wise multiplication between the two matrices. Following work [10], we scale $sigmoid(d_{ij})$ by 2, ensuring that its range falls within [0, 1] instead of [0, 0.5].

Second, pose temporal memory matrix $\boldsymbol{m}$ is transformed in feature space V and aligned to $\tilde{\boldsymbol{m}} \in \mathbb{R}^{n\times c}$ with the j^{th} row denoted as $\tilde{\boldsymbol{m}}_{\boldsymbol{j}}$,

$$\tilde{\boldsymbol{m}}_{\boldsymbol{j}} = \Sigma_{i=1}^{l} w_{i,j}(W_V \boldsymbol{m}_{\boldsymbol{i}}), \tag{2}$$

where $W_V \in \mathbb{R}^{c\times c}$ is learnable embedding weight.

Third, aligned memory $\tilde{\boldsymbol{m}}$ is added back to the original reshaped feature $\boldsymbol{x}$, and generated the temporal enhanced feature $\boldsymbol{y} \in \mathbb{R}^{n\times c}$, which is given by:

$$\boldsymbol{y} = \alpha\boldsymbol{x} + (1-\alpha)\tilde{\boldsymbol{m}}, \tag{3}$$

where α is a learnable scalar to balance $\boldsymbol{x}$ and $\tilde{\boldsymbol{m}}$. Finally, $\boldsymbol{y}$ is reshaped into the same shape as input feature map F, becoming the enhanced feature map $\tilde{F}$.

In the above formulation, W_K, W_Q, W_V are implemented as 1×1 convolutions. They transform the original features into a sub-space for similarity comparison. We use the multi-head attention [9, 11] in practice, which divides the embedding weights into N sets $\{W_K^1, ..., W_K^N\}$, $\{W_Q^1, ..., W_Q^N\}$ and $\{W_V^1, ..., W_V^N\}$, thus the correlation weights w computed by Eq. (1) becomes N sets $\{w^1, ..., w^N\}$, and $\tilde{\boldsymbol{m}}_{\boldsymbol{j}}$ in Eq. (2) also becomes $\tilde{\boldsymbol{m}}_{\boldsymbol{j}} = [\tilde{\boldsymbol{m}}_{\boldsymbol{j}}^{\boldsymbol{1}}, ..., \tilde{\boldsymbol{m}}_{\boldsymbol{j}}^{\boldsymbol{N}}]$ to compute the temporal enhanced feature $\boldsymbol{y}$ following Eq. (3).

***Write* operation** uses the quality of the generated heatmap to determine whether to store the corresponding temporal enhanced feature map $\tilde{F}$ into the pose temporal memory. It also has two stages.

In the first stage, we propose a novel method called keypoint feature replacement to refine the "feature-level" method to the "keypoint-level" method. It involves two steps. Firstly, the output heatmap score is computed to describe the detection quality for each keypoint. Similar to the process of transforming the heatmap into the final keypoint coordinates, the heatmap is first normalized to $[0, 1]$ and added 0.5 as the mean value. Then, a Gaussian filter is applied to blur the heatmap. The maximum value of the heatmap, whose position is considered as the highest possibility of the keypoint location, is set to the quality score of the corresponding keypoint. Secondly, based on the quality score of each keypoint, the feature regions corresponding to the low-quality keypoint locations would be replaced by previous high-quality feature regions. More specifically, if a keypoint score is higher than a high-quality threshold, this keypoint location in the heatmap would be mapped to the feature map region on the same scale between the heatmap and the feature map. Then the feature region is saved as a vector. If a keypoint score is lower than a low quality threshold which may affect the next frame detection adversely, the corresponding feature region would be replaced by the corresponding high quality keypoint feature region saved previously.

In the second stage, we sum all the keypoint scores as the heatmap score, which is used to determine if the corresponding feature is good enough to be put into the pose temporal memory. When the heatmap score is higher than a heatmap quality threshold, the selected temporal enhanced feature will be stored at new rows in the pose temporal memory matrix, which can make the future prediction more precise. To avoid too many features are loaded into the pose temporal memory, we set a capacity threshold for each pose temporal memory and delete the oldest feature from it when there is overloading.

3. EXPERIMENTS

3.1. Datasets and Evaluation

We perform experiments on the PoseTrack dataset[1]. The evaluation includes pose estimation accuracy and pose tracking accuracy. Following [1], we use the mAP metric and the clear MOT [12] metric to evaluate the multi-person pose estimation and tracking performance, respectively.

3.2. Ablation Study

Capacity of the pose temporal memory. We experiment to verify the relationship between the performance and the memory capacity, as shown in table 1. It is found that the performance would improve with the increase of the memory capacity, but it gains less improvement when the capacity is large. It is also found that the feature maps coming from the frames close to the current detecting frame have larger weights when aggregating with the current feature. Therefore, as we discussed in the *write* operation, we delete the oldest feature from the memory when there is overloading, because the old feature has less influence on the current feature enhancement.

Table 1: Comparison of different capacities in CPN backbone on PoseTrack'18 validation set.

capacity	total AP	total MOTA
1	72.1	62.7
2	72.6	63.0
4	72.9	63.2
8	73.1	63.4
16	73.2	63.4

Compositional De-Attention (CoDA). For better analysis, we use the average number of appearance people in a video to measure the degree of crowdedness. Videos with the average number over 8 are split from the validation set as a crowd subset. We test the two attentional models in the crowd subset. The results are given in Table 2.

From the table, it can be seen that in the crowd subset, CoDA is 2.1% and 1.3% higher on AP and MOTA than the vanilla attention. It proves that CoDA can explicitly learn the dissimilarity to better solve the occlusion in crowd scenes,

Table 2: Comparison of the different attentional models in CPN backbone on crowd subset.

method	total AP	total MOTA
vanilla attention	70.5	61.9
CoDA	**72.6**	**63.2**

while the vanilla attention is better at representing the similarity in uncrowded condition. Therefore, these two attentional models are combined to deal with different conditions for better performance.

Keypoint feature replacement. The keypoint feature replacement improves AP by 0.8% and MOTA by 0.5% based on the previous memory module. This strategy aims to deal with the extreme cases that the ordinary memory module cannot handle, while the dataset does not contain many of these situations. We believe this method will achieve more improvement in complex real-world scenarios.

Comparison with the baseline We compare the baseline top-down methods before and after adding-on our method on PoseTrack'18 validation set, as shown in Table 3. We get improvements of AP 3.8% and MOTA 2.9% using CPN backbone and AP 2.6% and MOTA 1.5% using SimpleBaseline backbone. All the other variable factors, such as the human detector, are kept the same in this experiment.

Table 3: Comparison with baseline methods on PoseTrack'18 validation set.

method	wrists AP	ankles AP	total AP	total MOTA
LightTrack-CPN	66.6	64.6	71.0	61.6
Ours-CPN	**69.0**	**65.1**	**74.8**	**64.5**
LightTrack-SimpleBaseline	65.6	64.6	72.4	63.8
Ours-SimpleBaseline	**66.6**	**66.2**	**75.0**	**65.3**

3.3. Memory usage in video pose estimation and tracking

The shape of the feature map stored in the pose temporal memory is (12, 9, 2048) using SimpleBaseline backbone and (12, 9, 256) using CPN backbone. Each feature map occupies no more than 100 MB in GPU memory. We can see in Table 1 that the promotion of the performance is less obvious with the increase of the memory capacity. Thus we set the memory capacity to 16 features in our experiment. The total memory occupancy is less than 9GB (i.e., about 7GB for the backbone and less than 2GB for our pose temporal memory module) using a single RTX 2080Ti with 11GB RAM.

3.4. Comparison with state-of-the-art results

We compare the state-of-the-art methods in the overall performance of the pose estimation and tracking task on PoseTrack'17 test and validation set, as shown in Table 4 and Table 5. We need to emphasize that the official code used on the evaluation server to evaluate the test set results has been changed at the beginning of 2020. It added a new constraint that the track ids in a single frame must be unique. Notice that the new requirement would decrease the final score especially for MOTA result, for a fair comparison, we have to re-run the baseline method LightTrack on test set after adding the new constraint, and only the previous SOTA methods' results on the validation set have comparability with our result.

Table 4: Comparison with LightTrack on PoseTrack'17 test set.

method	wrists AP	ankles AP	total AP	total MOTA
LightTrack (old evaluation)	64.6	58.4	66.7	58.0
LightTrack (new evaluation)	64.6	58.4	66.5	51.6
Ours (new evaluation)	64.2	61.3	69.9	53.0

Table 5: Comparison with the state-of-the-art methods on multi-person pose estimation and tracking on PoseTrack'17 validation set. The last column shows the speed in frames per second (* excludes pose inference time).

Method	Wrists AP / MOTA	Ankles AP / MOTA	Total AP / MOTA	FPS
Bottom-Up				
BUTD [15]	58.3 / 45.1	54.9 / 37.5	67.8 / 56.4	-
JointFlow [16]	- / -	- / -	69.3 / 59.8	0.2
CMU-Pose [17]	65.0 / -	62.7 / -	72.6 / 62.7	2
Top-Down				
ProTracker [4]	49.1 / 45.7	46.0 / 45.7	60.6 / 55.2	1.2
PoseFlow [18]	61.1 / 51.6	61.3 / 50.5	66.5 / 58.3	10*
FlowTrack [6]	72.4 / 56.1	67.1 / 53.5	76.7 / 65.4	-
LightTrack-CPN [5]	64.7 / 54.6	61.2 / 49.7	69.4 / 60.6	47* / 0.8
Ours-CPN	**67.3 / 57.0**	**62.4 / 52.3**	**73.5 / 63.6**	0.7
LightTrack-SimpleBaseline [5]	64.4 / 56.1	62.9 / 55.3	71.2 / 64.0	48* / 0.7
Ours-SimpleBaseline	**65.4 / 56.3**	**63.8 / 55.6**	**74.1 / 65.6**	0.6

Table 4 and Table 5 show that our method gets competitive results on PoseTrack and it only needs a little extra time and a few extra computing resources. Our method gets clear improvements of AP by 3.4% and MOTA by 1.4% on the test set, and AP by 2.9%, 4.1% and MOTA by 1.6%, 3.0% using SimpleBaseline and CPN backbone respectively on the validation set. We choose the single image-based human detector FPN [13] to generate human bounding boxes instead of using optical flow [14] in FlowTrack [6] which needs a lot of computing resources and is difficult to achieve the online effect. Besides, our module has no conflict with this optical flow method and it can also be transferred into the FlowTrack framework to get better performance.

4. CONCLUSION

In this paper, we propose the pose temporal memory module to better use the temporal information in the video pose estimation and tracking task. It is a general and light module, which is easy to be integrated into other top-down approaches. The designed *read* and *write* operations on the pose temporal memory can effectively and efficiently aggregate and store the temporal representation during the pose estimation and tracking process. The compositional de-attention (CoDA) and keypoint feature replacement are applied to respectively deal with the unique keypoint occlusion problem and recover the extreme error detection under fine-grained keypoint-level guidance. Moreover, we transfer our method to two widely used top-down backbones and improve the performance on the PoseTrack dataset, which verifies the generality and transportability of our module.

5. REFERENCES

[1] Mykhaylo Andriluka, Umar Iqbal, Eldar Insafutdinov, Leonid Pishchulin, Anton Milan, Juergen Gall, and Bernt Schiele, "Posetrack: A benchmark for human pose estimation and tracking," in *The IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[2] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun, "Cascaded pyramid network for multi-person pose estimation," in *The IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[3] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *The IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[4] Rohit Girdhar, Georgia Gkioxari, Lorenzo Torresani, Manohar Paluri, and Du Tran, "Detect-and-track: Efficient pose estimation in videos," in *The IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[5] Guanghan Ning and Heng Huang, "Lighttrack: A generic framework for online top-down human pose tracking," *arXiv preprint arXiv:1905.02822*, 2019.

[6] Bin Xiao, Haiping Wu, and Yichen Wei, "Simple baselines for human pose estimation and tracking," in *The European Conference on Computer Vision*, 2018.

[7] Hanming Deng, Yang Hua, Tao Song, Zongpu Zhang, Zhengui Xue, Ruhui Ma, Neil Robertson, and Haibing Guan, "Object guided external memory network for video object detection," in *The IEEE International Conference on Computer Vision*, 2019.

[8] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017.

[10] Yi Tay, Anh Tuan Luu, Aston Zhang, Shuohang Wang, and Siu Cheung Hui, "Compositional de-attention networks," in *Advances in Neural Information Processing Systems*, 2019.

[11] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei, "Relation networks for object detection," in *The IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[12] Keni Bernardin and Rainer Stiefelhagen, "Evaluating multiple object tracking performance: the clear mot metrics," *Journal on Image and Video Processing*, vol. 2008, pp. 1, 2008.

[13] Tsung-Yi Lin, Piotr Dollár, Ross B Girshick, Kaiming He, Bharath Hariharan, and Serge J Belongie, "Feature pyramid networks for object detection.," in *The IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[14] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *The IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[15] Sheng Jin, Xujie Ma, Zhipeng Han, Yue Wu, Wei Yang, Wentao Liu, Chen Qian, and Wanli Ouyang, "Towards multi-person pose tracking: Bottom-up and top-down methods," in *ICCV PoseTrack Workshop*, 2017.

[16] Andreas Doering, Umar Iqbal, and Juergen Gall, "Joint flow: Temporal flow fields for multi person tracking," in *British Machine Vision Conference*, 2018.

[17] Yaadhav Raaj, Haroon Idrees, Gines Hidalgo, and Yaser Sheikh, "Efficient online multi-person 2d pose tracking with recurrent spatio-temporal affinity fields," in *The IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[18] Yuliang Xiu, Jiefeng Li, Haoyu Wang, Yinghong Fang, and Cewu Lu, "Pose flow: Efficient online pose tracking," in *British Machine Vision Conference*, 2018.