ACTION SEGMENTATION ON REPRESENTATIONS OF SKELETON SEQUENCES USING TRANSFORMER NETWORKS

Simon Häring, Raphael Memmesheimer, Dietrich Paulus

University of Koblenz-Landau

ABSTRACT

We propose an approach for action segmentation by representing motions as images. A transformer object detection network is used to segment the sequences from the representation images. We examine different encoding approaches, normalization strategies and skeleton joint orders in an extensive experiment study. Our approach is evaluated on skeleton sequences from the PKU-MMD dataset. We successfully apply transformer networks for action segmentation on skeleton sequences. Our proposed approach achieves high class accuracies, while start and end-time estimation of the action segments are subject to further improvement.

***Index Terms*—** Action segmentation, Transformer, skeleton sequence, object detection

1. INTRODUCTION

The amount of video material recorded at any time has been growing due to inexpensive cameras and storage. Thus, automatic ways of processing and annotating video data became increasingly important. Human action recognition and its derivative tasks are one way to extract high-level information from recordings containing people. We focus on skeleton-based action recognition like [1, 2, 3, 4], which is supported by skeleton estimation methods including OpenPose [5] and RGB-D cameras like the Microsoft Kinect.

Surveillance and health-care could use systems that scan live video for suspicious actions or people in need of help. Scenarios like these include people falling or indicating sudden pain from stomach cramps or heart attacks by touching respective body parts. Further, human-robot-interaction can be improved by allowing robots to recognize actions continuously and act accordingly, i.e. to offer assistance. Action segmentation in particular allows for additional analysis of the action, like measuring its duration or the time between connected actions that belong to some longer action sequence.

We use an off-the-shelf network for object detection, namely the detection transformer DETR [6] and re-purpose it for human action segmentation. DETR addresses object detection with a simple model lacking hand-made components. Instead, it combines a CNN with a transformer network [7]. These attention-based transformers are designed

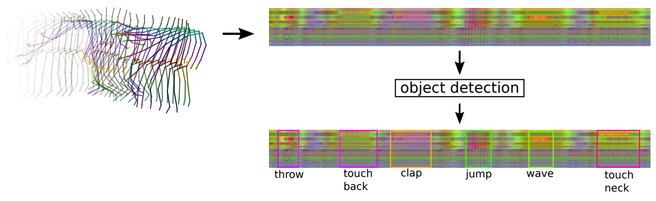

Fig. 1: Skeleton sequences are represented as images. A transformer network is trained to segment actions from the image representation.

to process long sequences and large sets, modeling arbitrary long-distance relations within. As opposed to Recurrent Neural Networks which maintain and update a memory vector used to reference information from earlier in the sequence, transformers act on the entire sequence at once and generate connections between different elements when they are necessary.

In order to use this existing network for human action segmentation we need to convert this task into object detection on images. We present and compare different ways of representing skeleton sequences as images. Unlike [2] and [4], we constrain ourselves to three channel or gray-scale images to keep compatibility with DETR and numerous other architectures. We use skeleton sequences instead of videos because they are more compact and therefore simplify processing long action sequences. The extracted skeletons are also invariant to changing background or lighting in the original video, which means however, that any contextual information is lost as well.

Our representations are constructed from different building blocks. These include normalization techniques, different coordinate encodings and image assembly methods like those in [1] or [8]. Our goal is to find best-performing combinations of the described techniques and provide an insight into which properties of a representation are beneficial for training. Figure 1 presents an example of our method. Different actions and movements appear as different texture patterns in the image representation. This is especially noticeable for the *jump* action in the middle of the image representation. The position and shape of each bounding box encodes the time-interval of the action. This is possible because time is encoded spatially in the image.

2. METHOD

We convert action segmentation on skeleton sequences into object detection on image representations.

2.1. Detection transformer - DETR

Object detection can be understood as a set prediction problem. An object detector generates a set of bounding boxes with class probabilities when given an image as its input. Just like sequences, sets can be predicted using transformers by evaluating multiple queries — one for each element in the output set. Carion et al. [6] present an end-to-end object detection pipeline based on the transformer architecture by Vaswani et al. DETR is built on top of a standard CNN for image data like ResNet50 or ResNet101 [9]. The 2048 feature maps generated by the CNN are reduced to 512 and flattened to a sequence of 512 dimensional vectors. A positional encoding vector of the same length is added to each input vector before the entire sequence is processed by the transformer. The sequence length is proportional to the image size. The encoder and decoder blocks of the transformer in DETR are the same as those presented by Vaswani et al. [7]. The input to the first decoder layer is a set of object queries which are unique, learned vectors. Each of these corresponds to one object prediction in the final output. The output vectors of this final decoder stage are independently transformed by a small feed-forward network into bounding box coordinates and class probabilities including a *no-object* class. The additional class is required as the number of object queries is constant and usually much larger than the expected number of objects in any single image. These object queries are a set of learned vectors. Unlike anchors or region proposals they do not explicitly encode a spatial region.

2.2. Representation

Skeleton sequences as they are generated by the Kinect v2 RGB-D camera contain up to six skeletons per frame. Each skeleton is defined by the 3D coordinates of its 25 joints. We construct our representation images by combining different techniques from the categories normalization, feature extraction and image assembly.

2.2.1. Normalization

In order to achieve location-invariance we normalize a skeleton sequence by shifting each skeleton such that its hip's center is the origin. We refer to this as **normPos** or **nP**. Alternatively we shift each skeleton such that the mean position of the hip joint is in the origin (**nPM**). Using this, we keep the movement information and only correct for fixed offsets. Scaling is done in a later step to constrain each feature to $[0, 255]$ for dense representations (see section 2.2.4). Similar to [3] and [4] we also use some of the joints to define a coordinate system relative to the mean skeleton of a sequence. The mean-hip defines the origin and the x-axis is given by a vector pointing from joint 17 to joint 13 in the pelvis. The z-axis is defined as the up-vector of the camera and the y-axis is $\vec{y} = \vec{z} \times \vec{x}$. All vectors are normalized. We will refer to this rotation-invariant normalization as **nRM**.

2.2.2. Feature extraction

Extracting additional information from skeleton sequences and explicitly providing it to neural networks can make them easier to train [2]. Other than the position **p** of the skeleton joints, we extract their velocity **v**, the angles **a** between their adjacent edges and the rates of change of those angles **A**. The angles are only generated for joints with two or more adjacent edges in a depth-first tree traversal order adapted from [1]. This leads to a total of 42 angles per skeleton. In the PKU-MMD dataset, two skeleton sequences are available for each video. If **both** are used, the representation images generated for each are stacked vertically.

2.2.3. Image assembly

We mainly focus on a dense image representation in which each column contains one frame such that time progresses from left to right. Each row holds one measurement of one joint, i.e. the y-velocity of joint five or the x-position of joint 23 etc. We order the joints either by their ID or by the tree structure described by Yang et al. [1]. The latter is referred to as **TSSI** in the following sections. Other than these dense representations, we use graphs to convert a skeleton sequence into an image like Memmesheimer et al. [8]. Here each coordinate of every joint is graphed in one combining image. These representations are denoted with **graph**.

2.2.4. Coordinate encoding

This section further details the way that the joint coordinates or velocities can be encoded in the dense representations shown above. An intuitive way of expressing three coordinates in a three-channel image is to encode each one in a separate color channel. We denote this as **RGB**, despite using OpenCV, which defaults to a BGR color order, to generate our representations. Figure 2 shows an example of a **nP RGB pv** skeleton representation of the action *clapping*. The upper half is a stack of the 3D positions of all 25 joints while the lower half contains their velocities. The rows of the first joint are black as its position and velocity is set to zero during the normalization (it is moved to the origin and stays there). When one-dimensional features like joint angles are added to this representation, each one is repeated to fill all three channels. Therefore, the resulting image contains a mixture of colored and gray-scale regions. Instead of stacking the (x, y, z) coordinates along the channel dimension, one could stack them along the joint dimension, forming a gray-scale image. We

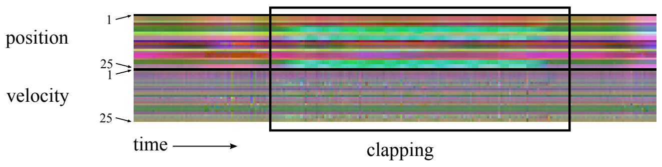

Fig. 2: A section of a **nP RGB pv** skeleton representation. The variation corresponds to a single clap, where the joint IDs related to the hand and arm contribute most to.

chose two orderings called **grayJ** $[x_1, y_1, z_1, ..., x_n, y_n, z_n]$ and **grayC** $[x_1, ..., x_n, y_1, ..., y_n, z_1, ..., z_n]$. The names stand for *gray* image with grouped *joints* and grouped *coordinates* respectively. While **RGB**, **grayC** and **grayJ** are mutually exclusive, either of them may be combined with **TSSI** in order to influence the skeleton joint order. In either case, including **RGB**, the values are mapped onto the range from $[0, 255]$ in order to fit into an 8-bit image. This is done separately for each type of feature, as the range of values for angles differ from that of positions or velocities.

3. EXPERIMENTS

In our experiments we show that an off-the-shelf object detector can be used for human action segmentation. We use the PyTorch [10] implementation of DETR, OpenCV for generating the representation images and the official PKU-MMD evaluation script for our results.

3.1. PKU Multi-Modality Dataset - PKU-MMD

PKU-MMD is a large-scale multi-modal dataset for human action recognition [11]. It contains 1076 untrimmed RGB, depth and infrared videos as well as skeleton sequences, captured by three Kinect v2 cameras from different viewpoints. Each three to four minute long sequence is annotated with a set of action labels containing start and end frames as well as one of 51 class labels. This makes the dataset ideal for action segmentation. Further, the actions are performed by 66 people and are split into two groups containing everyday actions and human-human interactions.

3.2. Implementation

Unless otherwise noted, we only use the first of possibly two skeletons available in the video for our experiments. We retrain the DETR weights provided in [6] with each representation on an Nvidia GTX 1080 for 2000 epochs with a learning rate of 3e-5, a class error weight of 8, a bounding box weight of 20, a DICE weight of 1, a relative weight of the no-object class of 0.1, a GIoU weight of 20 and no learning rate drop. Furthermore, we also use the cardinality loss, with weight 0.07, which compares the number of predicted objects to the actual number of objects in the image. For each image we let DETR predict 50 bounding boxes using its object queries. These hyper-parameters are kept constant unless further noted. All following mean average precision scores are percentages on the cross-view split of PKU-MMD grouping action instances either per video ($\text{mAP}_{\text{video}}$) or per action class ($\text{mAP}_{\text{action}}$). Like in the DETR implementation we use a ResNet-50 backbone which is trained alongside the transformer with a learning rate of 10e-5.

Table 1: Results for dense representations using different coordinate encoding methods.

Encoding	$\text{mAP}_{\text{action}}$	$\text{mAP}_{\text{video}}$
RGB	12.4	13.0
grayC	**45.1**	**46.1**
grayJ	28.7	29.5
3×**RGB**	44.6	45.0

Table 2: Results with the standard Kinect v2 and the TSSI joint orders and varying normalization methods.

	3×**RGB**		**grayC**	
Normalization	$\text{mAP}_{\text{action}}$	$\text{mAP}_{\text{video}}$	$\text{mAP}_{\text{action}}$	$\text{mAP}_{\text{video}}$
Kinect v2				
raw values	44.6	45.0	45.1	46.1
nP	**45.4**	**46.0**	42.1	41.7
nPM	39.0	40.4	47.6	48.7
nRM	40.4	40.6	**49.7**	**51.8**
TSSI				
nP	53.2	54.6	44.6	46.4
nRM	**56.3**	**58.0**	**51.3**	**52.4**

3.3. Results

We experimented with each of the previously presented components of our approach. Further we compare the dense representations against sparse representations and state-of-the-art methods.

In table 1 we present results for different dense representations of the skeleton position as described in section 2.2.4. The **grayC** encoding achieved best results. Comparably well is the 3 times stacked **RGB** representation, suggesting that the height width ratio of the final encoded image influences the results. We then compare different normalization strategies for the two better performing encodings **grayC** and 3×**RGB**. Table 2 shows the mean average precision scores for different normalization techniques using the standard Kinect v2 joint order (top) and the semantically sorted **TSSI** joint order (bottom). The **nP** had a positive effect on the Kinect v2 joint order, whereas the mean normalization for position and rotation resulted in a lower mAP. In contrast, for the **TSSI** joint order the **nRM** normalization yields the highest scores.

Table 3 shows results for different features combinations. Positions and angles perform much better than their time-

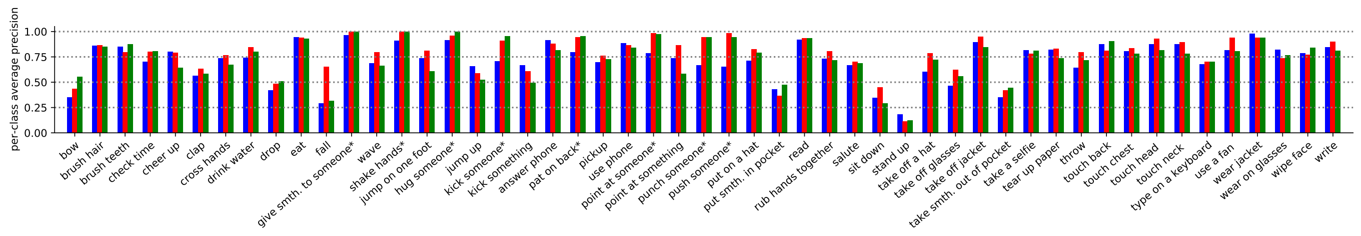

Fig. 3: The per-class AP for 4×**pa** (blue), 2×**pa both** (red) and hybrid (green). Classes marked with an asterisk are two-person interactions.

Table 3: Ablation Study modal features. Positional data (**p**) as before with versions using joint velocities (**v**), joint angles (**a**) and angular velocities (**A**) as well as stacks of these features.

		nRM TSSI 3×**RGB**	
Feature	height	$\text{mAP}_{\text{action}}$	$\text{mAP}_{\text{video}}$
p	147px	56.3	58.0
v	147px	22.9	21.6
a	126px	43.0	43.5
A	126px	9.7	10.1
pv	294px	67.2	68.1
pva	420px	**76.2**	**77.0**
pvaA	546px	70.2	71.4

Table 4: Results with two skeletons

Approach	$\text{mAP}_{\text{action}}$	$\text{mAP}_{\text{video}}$
4×**pa**	71.7	72.0
2×**pa both**	**77.9**	**79.0**
hybrid	73.6	74.7

derivatives. Combining the positional data with velocities and angles achieves the best results. However, adding angular velocities to this, leads to a reduction in mAP score. Disregarding the **pvaA** result, which may be explained by the low performance of angular velocities, we can observe the trend of taller representations resulting in better scores to continue.

In order to compare the influence of the second skeleton for interaction classes we test three representations built from stacks of positional and angular data. Figure 3 shows the per-class average precision of the three different representations discussed in this section. The two representations using both skeletons consistently score higher than 4×**pa** in interaction classes (marked with an asterisk). The large difference in mAP score between the hybrid and 2×**pa both** representations is mostly due to large improvements including *falling* and *pointing at something*. Table 4 shows the results of this experiment. While the 364px tall 4×**pa** representation scores in between the 294px 3×**pv** and 420px 3×**pva** representations from table 3, 2×**pa both** surpasses all previous representations by using the second skeleton where available. The hybrid representation does not achieve comparable results, despite also using both skeletons. Here, if only one skeleton is available, the image region is filled with a copy of the first skeleton instead of black pixels like with 2×**pa both**.

Table 5: Comparison of dense- and sparse representations.

Approach	height	$\text{mAP}_{\text{action}}$	$\text{mAP}_{\text{video}}$
graph p	420px	59.3	59.2
graph pa	420px	58.7	59.4
TSSI 6×**RGB p**	294px	68.0	69.5
TSSI 4×**RGB pa**	364px	**71.7**	**72.0**

Table 6: Comparison with related approaches.

Approach	mAP @ $\theta = 0.5$
Li et al. [12]	**94.4**
Li et al. [13]	93.7
JCRRNN [14]	53.3
nRM TSSI 3×**RGB pva** (ours)	77.0
nRM TSSI 2×**RGB pa both** (ours)	79.0

In order to show generalization of our action segmentation approach we integrate a recently proposed sparse **graph**-based representation for multi-modal action recognition [8]. In table 5 we compare two such **graph** representations with similar dense representations. Adding angular information to the **graph** representation seems to have no effect on the result. In table 6 we compare against related methods. Our approach is outperformed by the latest CNN-based approaches [12, 13] but performs better then the PKU-MMD dataset baselines[14]. None of the previous presented approaches uses transformer networks for the action segmentation task. While the class accuracy lies above 95% regularly on the test set, the GIoU, which determines how well our method predicts the start- and endpoints of an action, only reaches values around 75%.

4. CONCLUSION

We presented an approach for action segmentation on skeleton sequences, using a transformer-based object detector. We employ a variety of representations that can be used to flexibly encode skeleton motion into an image. The DETR approach is then used to segment action sequences from the PKU-MMD dataset. Our approach reaches a high class recognition accuracy but is outperformed by state-of-the art methods for skeleton-based action segmentation by lower start and end estimation of the actions.

5. REFERENCES

[1] Zhengyuan Yang, Yuncheng Li, Jianchao Yang, and Jiebo Luo, "Action recognition with spatio-temporal visual attention on skeleton image sequences," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 29, no. 8, pp. 2405–2415, 2019.

[2] Carlos Caetano, Jessica Sena de Souza, François Brémond, Jefersson A. dos Santos, and William Robson Schwartz, "Skelemotion: A new representation of skeleton joint sequences based on motion information for 3d action recognition," in *16th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2019, Taipei, Taiwan, September 18-21, 2019*. 2019, pp. 1–8, IEEE.

[3] Jian Liu, Naveed Akhtar, and Ajmal Mian, "Skepxels: Spatio-temporal image representation of human skeleton joints for action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2019, Long Beach, CA, USA, June 16-20, 2019*. 2019, Computer Vision Foundation / IEEE.

[4] Mengyuan Liu, Hong Liu, and Chen Chen, "Enhanced skeleton visualization for view invariant human action recognition," *Pattern Recognition*, vol. 68, pp. 346–362, 2017.

[5] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko, "End-to-end object detection with transformers," *CoRR*, vol. abs/2005.12872, 2020.

[7] 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 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, Eds., 2017, pp. 5998–6008.

[8] Raphael Memmesheimer, Nick Theisen, and Dietrich Paulus, "Gimme signals: Discriminative signal encoding for multimodal activity recognition," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA, 2020, IEEE.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. 2016, pp. 770–778, IEEE Computer Society.

[10] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, Eds., pp. 8024–8035. Curran Associates, Inc., 2019.

[11] Chunhui Liu, Yueyu Hu, Yanghao Li, Sijie Song, and Jiaying Liu, "PKU-MMD: A large scale benchmark for skeleton-based human action understanding," in *Proceedings of the Workshop on Visual Analysis in Smart and Connected Communities, VSCC@MM 2017, Mountain View, CA, USA, October 23, 2017*, Xiaobai Liu, Yadong Mu, Yu-Gang Jiang, and Jiebo Luo, Eds. 2017, pp. 1–8, ACM.

[12] Tianhong Li, Lijie Fan, Mingmin Zhao, Yingcheng Liu, and Dina Katabi, "Making the invisible visible: Action recognition through walls and occlusions," in *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*. 2019, pp. 872–881, IEEE.

[13] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu, "Skeleton-based action recognition with convolutional neural networks," in *2017 IEEE International Conference on Multimedia & Expo Workshops, ICME Workshops, Hong Kong, China, July 10-14, 2017*. 2017, pp. 597–600, IEEE Computer Society.

[14] Yanghao Li, Cuiling Lan, Junliang Xing, Wenjun Zeng, Chunfeng Yuan, and Jiaying Liu, "Online human action detection using joint classification-regression recurrent neural networks," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VII*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, Eds. 2016, vol. 9911 of *Lecture Notes in Computer Science*, pp. 203–220, Springer.