TAD-Net: An approach for real-time action detection based on temporal convolution network and graph convolution network in digital twin shop-floor [version 1; peer review: awaiting peer review]

Qing Hong\(^1\), Yifeng Sun\(^2\), Tingyu Liu\(^3\), Liang Fu\(^1\), Yunfeng Xie\(^1\)

\(^1\)School of Mechanical Engineering, Nanjing University of Science and Technology, Nanjing, 210094, China
\(^2\)China Electronics Technology Group 28th Research Institute, Nanjing, 210007, China
\(^3\)School of Mechanical Engineering, Southeast University, Nanjing, 210096, China

**Abstract**

**Background:** Intelligent monitoring of human action in production is an important step to help standardize production processes and construct a digital twin shop-floor rapidly. Human action has a significant impact on the production safety and efficiency of a shop-floor, however, because of the high individual initiative of humans, it is difficult to realize real-time action detection in a digital twin shop-floor.

**Methods:** We proposed a real-time detection approach for shop-floor production action. This approach used the sequence data of continuous human skeleton joints sequences as the input. We then reconstructed the Joint Classification-Regression Recurrent Neural Networks (JCR-RNN) based on Temporal Convolution Network (TCN) and Graph Convolution Network (GCN). We called this approach the Temporal Action Detection Net (TAD-Net), which realized real-time shop-floor production action detection.

**Results:** The results of the verification experiment showed that our approach has achieved a high temporal positioning score, recognition speed, and accuracy when applied to the existing Online Action Detection (OAD) dataset and the Nanjing University of Science and Technology 3 Dimensions (NJUST3D) dataset. TAD-Net can meet the actual needs of the digital twin shop-floor.

**Conclusions:** Our method has higher recognition accuracy, temporal positioning accuracy, and faster running speed than other mainstream network models, it can better meet actual application requirements, and has important research value and practical significance for standardizing shop-floor production processes, reducing production security risks, and contributing to the understanding of real-time production action.
Keywords
Digital twin shop-floor, Production action, Real-time action detection, TAD-Net, TCN, GCN

Corresponding author: Tingyu Liu (tingyu@seu.edu.cn)

Author roles: Hong Q: Data Curation, Formal Analysis, Validation, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing; Sun Y: Software, Validation, Writing – Original Draft Preparation; Liu T: Conceptualization, Project Administration, Resources, Supervision; Fu L: Investigation, Methodology, Validation; Xie Y: Investigation, Software, Visualization

Competing interests: No competing interests were disclosed.

Grant information: This work was supported by the National Key Research and Development Program, China (2020YFB1708400) and the National Defense Fundamental Research Program, China (JCKY2020210B006, JCKY2017204B053) awarded to TL.

Copyright: © 2021 Hong Q et al. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

How to cite this article: Hong Q, Sun Y, Liu T et al. TAD-Net: An approach for real-time action detection based on temporal convolution network and graph convolution network in digital twin shop-floor (version 1; peer review: awaiting peer review)
Digital Twin 2021, 1:10 https://doi.org/10.12688/digitaltwin.17408.1

First published: 01 Dec 2021, 1:10 https://doi.org/10.12688/digitaltwin.17408.1
Introduction

Shop-floor is the key unit for the manufacturing industry, and its digitization and intelligence are necessary to realize intelligent manufacturing\(^1\). The proposal of the digital twin shop-floor and its application provide effective theoretical and technical support for the efficient management and operation of shop-floor production processes\(^1\). Digital twin technology aims to construct a deep integration and mapping between the physical world and virtual world by digital means, such as multi-dimensional and multi-scale virtualization of physical entities, and aims to explore the simulation, analysis and optimization of physical entities and their actions based on virtual models\(^2\). To realize the intelligent control of manufacturing shop-floors, digital twin shop-floor modeling is an indispensable step\(^3\).

Compared to equipment, products and other production elements, humans are important elements in areas such as production design and manufacturing activities. Therefore, human action has always been the focus in controlling shop-floor activities\(^4\). Complex electromechanical product shop-floors usually have large scale, multi-station, complex environments and dangerous key processes. The uncertainty of human action has a significant impact on shop-floor production security and efficiency\(^4\). Traditional video or sensor surveillance methods cannot realize intelligent monitoring of human action. For example, data acquisition sensors on workers cannot easily be applied in complex production processes\(^5\). At this time, the digital description of human action has the potential to provide services such as efficient monitoring of human-computer interaction\(^6\) and human controlling\(^6\) in digital twin shop-floors, thus improving production safety levels and efficiency of shop-floors. Therefore, to realize human production actions’ intelligent control in digital twin shop-floors, it is necessary to propose an effective approach to detect these actions, and thus to reach the target of intelligent monitoring of humans in digital twin shop-floors\(^6\).\(^7\).

Human action detection differs from human action recognition. The purpose of action recognition is to classify a video or sequence segment containing single action\(^8\)-\(^10\), and the purpose of action detection is to locate and classify a video or sequence containing several unknown actions in time domain\(^10\)-\(^12\). Therefore, action recognition is the basis of action detection tasks. Due to the high subjective initiative and uncertainty of humans, their actions are difficult to accurately detect, and the accuracy of traditional action recognition methods with manual design features is poor in practical shop-floor applications. It is therefore difficult to meet the application requirements of production action recognition\(^6\).\(^7\). Therefore, the current methods to realize human action recognition are mostly based on deep learning algorithms, and their digital expression carriers of human actions are mainly image videos\(^1\), vector sequence\(^2\)-\(^10\) or topological sequence\(^6\).\(^9\). However, image video data has huge information redundancy, and those action recognition algorithms based on this data consume massive arithmetic resources. It is difficult to apply in practice\(^6\).\(^7\). Therefore, the current research on action recognition mostly uses the human skeleton joints sequence with low complexity as the input data. For example, 7 constructed an action’s spatio-temporal feature map based on human skeleton joints sequence, achieving good results in production action recognition task. Reference 6 constructed a human twin-model based on skeleton joints, and proposed a shop-floor production action recognition method based on the Graph Convolution Network (GCN), which greatly improved the classification accuracy of production action. However, these kinds of action recognition methods can only identify a temporal sequence data containing a single action, and cannot process the real-time continuous shop-floor data stream. They can be applied to identify the segment only after these action segments were extracted from the real-time data stream\(^10\)-\(^12\). Therefore, the key to shop-floor human’s real-time action detection is to realize the action temporal position first, and then to realize action recognition.

Action time domain detection is similar to object detection\(^3\)-\(^14\) in computer vision, and belongs to the visual understanding field. The difference is that the final result of action time domain detection is to get the action segment in video or sequence data, instead of in object boundary box in a picture\(^10\)-\(^19\). Compared to object detection, action time domain detection is more difficult, mainly due to the following\(^10\)-\(^19\):

(1) Timing information. The monitor data containing temporal information is huge, and the difficulty of action detection is to effectively extract and process feature information in these complex redundant data.

(2) Unclear action boundary. Action detection needs to accurately position the starting and ending time of every action, and the action boundary in manufacturing is often unknown, resulting in difficult application of real-time action detection in real shop-floors.

(3) Large time span. Due to the uncertainty of action in shop-floor manufacturing, the action time span is very large. It is as short as a few frames or as long as hundreds of frames, making it extremely difficult to design candidate timing regions.

The implementation process of action detection task is similar to object detection\(^3\)-\(^14\). First, the action’s candidate timing region is obtained (that is, to determine the start and end time of an action in the time domain), and then these sequence segments are recognized\(^10\)-\(^19\). For example, 15 obtained candidate action region by the sliding window method\(^11\),\(^12\), and used Support Vector Machine (SVM) to classify these action segments, and thus to obtain the categories of these action. However, this kind of approach is extremely inefficient because multiple sliding windows with different lengths are used at every time point in the video, resulting in extremely low efficiency. At the same time, if the sliding window is not dense enough, it will lead to inaccurate boundary positioning. Therefore, many subsequent action detection approaches are based on these image object detection algorithms\(^13\),\(^14\) with better performance. The main idea was to design the time domain...
candidate box to extract end-to-end action segments in video or sequence data, and to classify them. For example, the Region Convolutional 3D (R-C3D) network was proposed in [16] based on the Faster Regions with Convolutional Neural Network features (Faster R-CNN) framework. In R-C3D, the video frames in three-dimensional convolution network were encoded, action candidate segments were extracted by designing several time domain candidate boxes, and recognition of action segments in classification sub-network was realized. Similarly, the Structured Segment Network (SS-Net) and Convolutional-De-Convolutional (CDC) network models were proposed. Although these algorithms can detect any length of action segments, they are too dependent on the size of the preset candidate box, and their convolution operation for video frames requires huge computing resources, resulting in low running speed and poor generalization ability. To reduce the consumption of computing resources, based on human skeleton joints sequence flow with lower complexity, proposed an action detection algorithm based on Long Short Term Memory (LSTM), and judged action category in every frame of the sequence flow. However, its recognition accuracy for more complex action is not good. In order to improve recognition accuracy, proposed an action detection approach based on LSTM, added the design of joint classification and regression, and constructed a Joint Classification-Regression Recurrent Network (JCR-RNN), which greatly improved the accuracy of complex action recognition. However, due to the limitation of the LSTM network itself, the detection speed was still slow, and failed to meet the real-time requirements of human production action detection in digital twin shop-floor.

In view of this, we proposed a real-time detection approach for shop-floor production action. This method used continuous skeleton joints sequence data as the input, and strengthened JCR-RNN’s action time domain positioning network based on Temporal Convolution Network (TCN), which greatly improved the accuracy of complex action recognition. However, due to the limitation of the LSTM network itself, the detection speed was still slow, and failed to meet the real-time requirements of human production action detection in digital twin shop-floor.

**Methods**

**Ethics statement**

No ethical approval was sought for the study because the research was low-risk, non-interventional, did not collect any identifying data from the participants (only joint sequence data were collected). All participants involved in the data collection provided their written informed consent (via signature) to participate in the study and for the data associated with the study to be published.

**Technical route**

The technical route of real-time shop-floor production action detection is as follows:

(1) Input data acquisition and preprocessing: Collect production action skeleton joints sequence data by Kinect (V2) depth vision sensor in shop-floor station, and reconstruct and correct these skeleton joints sequence data based on preprocessing method (translation, rotation, normalization) outlined in [6,20]. The custom code for the preprocessing method (gendata.rar) is provided in Underlying data. We called the interface function programs (‘rotation_matrix’, ‘transformation’) in the ‘gendata.rar’ source code to complete the preprocessing of human skeleton joints sequence.

(2) TAD-Net constructing and training: Enhance temporal extraction module of JCR-RNN by replacing LSTM with TCN, process timing information of input data, and extract its timing features. Based on the methods outlined in GCN, reconstruct JCR-RNN’s regression module and classification module instead of ordinary convolution network, output the starting and ending time of action using the regression module (only output the starting time for real-time detection), and output action category using the classification module. Combine these two modules to form our Temporal Action Detection Network (TAD-Net), and train it based on the Online Action Detection (OAD) and Nanjing University of Science and Technology (NJUST3D) datasets (see Underlying data).

(3) TAD-Net verification: Based on the long sequence action detection dataset OAD and the shop-floor production action dataset NJUST3D, verify TAD-Net by comparing with other mainstream networks, and calculate their Start Localization Score (SL-Score), classification accuracy, and Frames Per Second (FPS). Compare this with other approaches to analyze and verify the feasibility of our real-time detection approach for shop-floor production action. Based on these experimental results, discuss and point out the existing problems and improvement directions.

**Temporal Action Detection Network (TAD-Net)**

The overall structure of time domain action detection network (TAD-Net) is similar to JCR-RNN. As shown in Figure 1, TAD-Net mainly includes three network modules: a timing extraction module, a classification module and a regression module. Based on TCN, JCR-RNN’s original LSTM timing extraction module was enhanced, and based on GCN, the original ordinary convolution classification and regression module were reconstructed. The timing feature of continuous skeleton joints sequence data was extracted by the timing extraction module, and was transmitted to the classification and regression module in the form of temporary matrix data, then the timing feature information was processed using the graph convolution operation, and the real-time action classification results were output using the classification module. The action starting times were output using the regression module (see the Timing extraction module, Classification module, and Regression module sections for full methods). The specific design of each module is as outlined in the following sections.

**Timing extraction module.** TCN is a network structure to extract timing features from sequence/video data. Most scholars’ research on TCN shows that it performs better than those...
kinds of recurrent neural networks such as LSTM in timing information extraction tasks. Therefore, TCN structure was selected instead of LSTM as the key module for extracting production action timing features information.

The single TCN block network structure designed in TAD-Net is shown in Figure 2. Inspired by Natural Language Processing (NLP) tasks, our TCN block uses both residual and parameterized skip connections, each layer uses dilated convolution and then uses residual structure to alleviate gradient disappearance and explosion problems. Therefore, our TCN block can be stacked in multiple layers to form a more complex timing extraction network to get more features from longer span actions. To better alleviate the gradient disappearance problem, the gated activation function is used in our TCN network. This function is shown in Formula (1):

$$z = \tanh (\omega_{f,k} \ast x) \odot \sigma(\omega_{g,k} \ast x)$$  \hspace{1cm} (1)

In Formula (1): $\ast$ is the convolution operation, $\odot$ is the matrix dot multiplication, $\sigma$ is the result of sigmoid activation function, and $\omega$ is the convolution kernel parameter, $\tanh$ is the tanh activation function, $x$ is the input value.

In order to solve the problem of different lengths of production action segments, our timing extraction module adopts a timing convolution structure similar to the hierarchical pyramid, so that the network can obtain the most suitable receptive field by selecting different convolution layers, and can focus on those ongoing action timing parts. This helps in avoiding timing feature loss due to excessive useless information. Taking 3-layer-TCN timing extraction network module as an example, the network’s internal information transmission is shown in Figure 3. The higher layer will further extract timing information based on the feature information extracted by the lower layer, to get a larger receptive field.

At the same time, the single TCN block of our timing extraction module uses dilated convolution to further increase the receptive field and strengthen the TCN block’s processing ability for longer-duration production actions. To test the receptive field of TCN block, we collected action segments at...
30 frames per second and recorded the longest frame/time that n-layer TCN can extract effective information.

In this experiment, one of our authors (YS) performed production actions in the shop-floor at six different durations (32, 64, 128, 256, 512, 1024fps). We collected their skeleton joints sequence data by a Kinect (V2) sensor. Each data was used as input data separately to test whether each layer of TCN can completely extract the feature information.

The receptive field values of the 5, 6, 7, 8, and 9-layer TCN timing extraction networks are shown in Table 1. The receptive field values of the collected data are calculated at 30 frames per second. For example, 8-layer network’s receptive field is 256 frames, that is 8.53s, and 9-layer network’s field is 17.07s. Therefore, the number of layers can be flexibly selected according to the frames of longest action, making it suitable for the current action.

**Classification module.** Considering that the human skeleton joints are rigid structures similar to hinges, these joints can be connected as a topological structure based on the correlation of human joints. Unlike with direct coordinate connection, topological connection can highlight human skeleton joints’ spatial posture features more effectively. With the Graph Convolution Network (GCN), the spatial features of human skeleton joints can be extracted more effectively, thus increasing the accuracy of action recognition. Therefore, we built a new classification network based on GCN to replace JCR-RNN’s original ordinary convolutional network.

Based on GCN’s stronger feature extraction ability in graph nodes aspects, it can be applied to the action recognition of human skeleton joints. Take the human skeleton topological graph \( G = \{\text{joints}, \text{bones}\} \) as an example, the graph structure \( G \) is the combination of the vertices in \( \text{joints} = \{V_1, \ldots, V_n\} \) and the sides in \( \text{bones} \subseteq \text{joints} \times \text{joints} \), where \( V_i = \{x_i, y_i, z_i\} \), represents the three-dimensional coordinate data of each joint. The sides in \( \text{bones} \) are human body skeletons. The graph convolution can be defined as:

\[
    f_{\text{out}}(v_g) = \sum_{v_i \in \text{B}(v_g)} \frac{1}{Z_{\text{g}}(v_g)} f_m(v_g) \cdot w(t_i(v_g))
\]

In Formula (2): \( f \) is the feature map, \( v \) is the graph vertex, \( B(v_g) \) is the mapping of vertex \( v_g \) to all adjacent vertices, \( Z_{\text{g}}(v_g) \) is the number of adjacent vertices, \( t_i(v_g) \) is the mapping of vertex \( v_g \) to adjacent vertices. \( w \) is a weighting function similar to ordinary convolution operation, which provides a weight vector based on a given input. However, the number of weight vectors for ordinary convolution operations is fixed, while the number of vertices in \( B(v_g) \) is variable, so we let \( B(v_g) \) divide by \( Z_{\text{g}}(v_g) \) to eliminate this uncertainty influence.

The single GCN block network structure in our TAD-Net is shown in Figure 4. Embedding the attention module enables the GCN block to pay attention to different joints when recognizing different actions, and thus can dynamically assign different attention weights for each joint. The attention mechanism includes the encoding and decoding layers. After the decoding layer, the softmax function is added to

---

[Figure 3. Three-layer temporal convolution network timing extraction network information transmission. d, the hyperparameter of dilated convolution; t, the current time frame; C, the characteristic data of current network layer.]
make the sum of all joint attentions 1, and the residual structure is added to prevent gradient explosion. The attention mechanism formula in our GCN is shown in Formula (3):

$$f_{out} = \sum_{k}^{K} W_k f_{in} (A + softmax(f_{in}^T W_k^T W_k f_{in}))$$  

In Formula (3): $A$ is the joints adjacency matrix, $W_k$ is the graph convolution operation, $f_{in}$ is the input feature, $f_{out}$ is the output feature.

The common features of graph vertices and their respective weights were determined by the $n \times n$ matrix parameters, which are output by softmax function. The Normalized Embedding Gaussian function was used to calculate the similarity of the two vertices.

Each TCN block layer’s results were combined at each time point. These features were then processed by the classification module to obtain the result of the action’s real-time classification. The classification network module first uses a multi-layer GCN to construct multi-level structured features, then adds a Full Connection (FC) layer and a softmax layer to classify the current action segment, and finally outputs the probability distribution of each action.

**Regression module.** In object detection task, the general object positioning in the image is located by a bounding box. The bounding box is generally represented by a four-dimensional vector $(x, y, w, h)$, with $x, y$ representing the bounding box center coordinate in the picture pixel, $w, h$ represent the pixel width and height of the bounding box. This is different for the action detection task; the skeleton joint action sequence data can be regarded as one-dimensional data in a time domain. That is, the bounding box only needs two vectors to represent the starting and ending time. The real-time action detection is also different because of real-time data flow’s continuity. If the bounding box is expressed in absolute coordinates, then as time goes by, the starting time and ending time of each action will continue to increase, causing inconvenience in calculation. Therefore, we select a bounding box based on relative coordinates. The current time coordinate is set as the difference between current time and current behavior starting time, that is, the relative coordinate of the current time.

The Mean Square Error loss function is used during training to estimate errors between predicted data and original data, and its formula is as shown in Formula (4):

$$MSE = \frac{1}{n} \sum_{i=1}^{m} (s_i - \hat{s}_i)^2$$  

In Formula (4): $s_i$ is the real label, and $\hat{s}_i$ is the model prediction data.

Based on the construction of the above classification module, and considering the continuity and the real-time nature of actual shop-floor data, it is necessary to detect the action starting time in real-time data flow, and because we cannot know the ending time of real-time action, the real-time regression task is different from other offline action detection tasks. It only outputs the relative coordinates for the current action starting time, and sets the ending time while the output action category of the classification network changes.

<table>
<thead>
<tr>
<th>Input Frame/ Network</th>
<th>TCN-5</th>
<th>TCN-6</th>
<th>TCN-7</th>
<th>TCN-8</th>
<th>TCN-9</th>
</tr>
</thead>
<tbody>
<tr>
<td>30fps</td>
<td>1.07s/32fps</td>
<td>2.13s/64fps</td>
<td>4.27s/128fps</td>
<td>8.53s/256fps</td>
<td>17.07s/512fps</td>
</tr>
</tbody>
</table>

**Table 1. The relationship between the model layer number and receptive field.** TCN, Temporal Convolution Network; TCN-5, the 5-layer Temporal Convolution Network; fps, frames per second.
Implementation and operation requirements. The code associated with TAD-Net is provided in Software availability. The minimum hardware requirements to run our TAD-Net deployment are shown in Table 2, and the corresponding software that TAD-Net relies on are shown in Table 3.

Algorithm verification and analysis
Our experimental verification is carried out based on Python 3.5 and Windows 10 software systems, NVIDIA GTX1080Ti hardware and the open source software NVIDIA CUDA 10.0 Toolkit GPU accelerated environment. In order to verify the accuracy and reliability of TAD-Net, we conduct action recognition accuracy tests, temporal positioning accuracy tests, and running speed tests for real-time shop-floor production actions, as outlined in 10.17,33. These tests were conducted by comparing TAD-Net to other mainstream network models, namely the long sequence action dataset OAD and the shop-floor production action dataset NJUST3D.

Test datasets. The OAD (Online Action Detection) dataset (see Underlying data) is an open source, long sequence of action detection dataset collected for online action detection problems. It is collected daily in an indoor environment by Yanghao Li’s team. In this dataset, different actors performed 10 actions freely, such as drinking, eating, writing, opening a cupboard, hand washing, opening a microwave, and sweeping. The actions were captured using a Kinect (V2) sensor, which collects color images, depth images and human skeleton joints synchronously. The dataset includes 59 long sequences and 10 actions.

The NJUST3D (Nanjing University of Science and Technology 3 Dimensions) dataset (see Underlying data) is an open source, continuous shop-floor production action skeleton joints sequence dataset collected in the actual shop-floor environment by our team. This dataset was previously published (we have its total copyright). Compared to the OAD dataset, it has a higher data frame rate, more categories of actions and longer action duration, and is therefore closer to the reality of the shop-floor. It was captured using the Kinect (V2) sensor, which collects human skeleton joints. The dataset includes 13 ultra-long-frame sequences and 18 production actions.

Test methods
(1) Action recognition accuracy
To evaluate the action recognition accuracy of TAD-Net on datasets, we tested existing mainstream network models (namely the Spatio-temporal Action Long Short Term Memory (STA-LSTM), the Joint Classification-Regression Recurrent Network (JCR-RNN), and the Structured Segment Network (SSNet)) and our TAD-Net separately on the OAD and NJUST3D dataset. The test steps are as follows: randomly divide OAD and NJUST3D datasets into a train-set and a test-set according to 4:1. Train the network models on the training set for 100 epoch in their own approach, and then test on the test-set in the same way (count the number of total test-set and correct predictions) and obtain the number of total test-set and correct predictions. Finally, divide the number of correct predictions by the total number of test-set. This result is the average accuracy rate of this model. We draw the comparison accuracy curve of each model according to these test results.

In order to more intuitively show the recognition accuracy and the confusion degree between various actions, we draw the recognition accuracy confusion matrix (a table to fully view the confusion relationship of all kinds of actions) of TAD-Net on the dataset based on its recognition accuracy.

(2) Temporal positioning accuracy
To evaluate the positioning accuracy of action starting and ending time, we use the Start Localization Score (SL-score) based on the relative distance between predicted action starting time and real time. Assuming that algorithm predicted action starting time is $t_{start}$ and the action’s corresponding real action interval is $[t_{start}, t_{end}]$, SL-score calculation formula is as shown in Formula (5):\[ SLScore = e^{-\frac{(t_{end} - t_{start})^2}{2\sigma^2}} \] For incorrect predicted samples, their SL-Score is set to 0. In addition, the definition of action boundary caused by manual labeling is blurred, and there is less information at the

<table>
<thead>
<tr>
<th>Equipment</th>
<th>Name</th>
<th>Minimum requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer</td>
<td>CPU</td>
<td>Intel i5 6500</td>
</tr>
<tr>
<td></td>
<td>GPU</td>
<td>Nvidia GeForce 1060</td>
</tr>
<tr>
<td></td>
<td>Memory</td>
<td>8G</td>
</tr>
<tr>
<td></td>
<td>GPU Memory</td>
<td>4G</td>
</tr>
<tr>
<td></td>
<td>Hard disk memory</td>
<td>1T</td>
</tr>
<tr>
<td>KinetV2</td>
<td>Operation system</td>
<td>Windows10</td>
</tr>
<tr>
<td></td>
<td>Interface</td>
<td>USB3.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Software and version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation system</td>
<td>Windows10 x64</td>
</tr>
<tr>
<td>Development tool</td>
<td>Visual Studio 2015, Python 3.5.6</td>
</tr>
<tr>
<td>Coding language</td>
<td>C#, Python, C++</td>
</tr>
<tr>
<td>Dependent libraries</td>
<td>Pytorch 1.1, OpenCV 3.4.0, Kinect V2</td>
</tr>
<tr>
<td>Databases</td>
<td>Oracle 11XE, MySQL 8.0.14</td>
</tr>
<tr>
<td>Accelerated environment</td>
<td>CUDA 10.0</td>
</tr>
</tbody>
</table>
action’s initial stage, therefore, we reserve 0.5s as each action’s starting blur point, and regard it as a background point.

Similar to the test of recognition accuracy, we test the temporal positioning performance of each model based on SL-Score on the test-set, and draw the comparison curve.

(3) Running speed for real-time action

Due to fact that the digital twin shop-floor requires high running speeds for real-time action detection algorithms, and in order to verify the real-time performance of these mainstream models, we used FPS as an indicator, and measured the actual speed of each algorithm based on real-time data flow in the shop-floor.

The specific steps are as follows: firstly, we designed the sequential script for the workers to perform the production action. The classes of production actions are consistent with the NJUST3D dataset, the duration of each action is controlled between 1 and 15 seconds. Then, in the shop-floor, seven participants including four of the authors of this study (QH, YS, LF, YX) performed a total of 480 different production actions in order (different durations, different action types, and different workers). We obtained the skeleton joints sequences data of these actions by Kinect (V2) sensor (only the three-dimensional coordinates of 25 joint points of human body were recorded, excluding any personal information of the participants). We identified the real-time action data based on each model, recorded the total time spent on the 500 actions’ recognition, and divided the total frames of the 500 actions by this total time data to obtain the model FPS, and thus we were able to judge the real-time running speed of the model, and draw the comparison curve (see Underlying data20).

Results

Action recognition accuracy

Based on OAD dataset, we tested action recognition accuracy on mainstream network models STA-LSTM33, JCR-RNN10, SS-Net17 and our TAD-Net with different network layers, and their tested average recognition accuracy results are shown in Figure 5. From this figure, it can be seen that TAD-Net performed better than other mainstream models in terms of accuracy.

To verify the recognition confusion degree of TAD-Net for different actions, we drew the recognition confusion matrix of TAD-Net-6 on OAD dataset, as shown in Figure 6. Its vertical axis is real category and horizontal axis is prediction category. On this confusion matrix, TAD-Net-6 has a higher confusion probability for actions with similar skeleton joints, such as drinking and eating, and has lower confusion probability for dissimilar actions.

Similar to the testing processes on OAD dataset, the recognition accuracy obtained based on NJUST3D dataset test are shown in Figure 7. From the accuracy, it can be seen that TAD-Net obviously performed better than other mainstream models, and it is more suitable for actual shop-floor production action datasets.

To verify the recognition confusion degree of TAD-Net for various actual production actions, we drew the recognition confusion matrix of TAD-Net-9 on NJUST3D dataset. As shown in Figure 8, its vertical axis is real category and horizontal axis is predicted category. On this confusion matrix, similar to matrix on OAD dataset, TAD-Net-9 has a higher confusion probability for similar production actions, perform better for actions with dissimilar skeleton joints.

Temporal positioning accuracy

To verify the action temporal positioning ability of each model, we evaluated the action starting time regression module of these models, calculated their SL-Score based on OAD and NJUST3D dataset, and the results are shown in Figure 9 and Figure 10 respectively. The SL-Score results comparison shows that TAD-Net has better performance in action temporal positioning task.
Running speed for real-time action

The test results of models’ average running speed are shown in the Figure 11. It can be seen that TAD-Net can reach 20fps under the premise of meeting receptive field requirements, and TAD-Net’s running speed in actual shop-floor is higher than other models.

Discussion

Based on OAD and NJUST3D dataset experiment results, the performance of each mainstream model was analyzed:

1. In terms of recognition accuracy, because video data have far more information than human skeleton joints sequence.
Figure 8. Confusion matrix of TAD-Net-9 on NJUST3D Dataset. NJUST3D, the Nanjing University of Science and Technology 3 Dimensions; TAD-Net-9, the 9-layer Temporal Action Network.

Figure 9. SL-Score of each model on the Online Action Detection (OAD) dataset. SL-Score, the Start Localization Score; STA-LSTM, the Spatio-temporal Action Long Short Term Memory; JCR-RNN, the joint Classification-Regression Recurrent Network; SSNet, the Structured Segment Network; TAD-Net-5, the 5-layer Temporal Action Network.

data, SS-Net based on video flow data performed better than STA-LSTM and JCR-RNN based on skeleton joints sequence flow data. However, GCN has a stronger ability to extract the inherent correlation features in skeleton joints than ordinary network\(a\), without the interference of the video’s redundant background factors. The recognition accuracy of TAD-Net
Figure 10. SL-Score of every model on the Nanjing University of Science and Technology 3 Dimensions (NJU3T3D) dataset. SL-Score, the Start Localization Score; STA-LSTM, the Spatio-temporal Action Long Short Term Memory; JCR-RNN, the Joint Classification-Regression Recurrent Network; SSNet, the Structured Segment Network; TAD-Net-5, the 5-layer Temporal Action Network.

Figure 11. Model average running speed. FPS, Frames Per Second; STA-LSTM, the Spatio-temporal Action Long Short Term Memory; JCR-RNN, the joint Classification-Regression Recurrent Network; SSNet, the Structured Segment Network; TAD-Net-5, the 5-layer Temporal Action Network.

based on GCN is slightly higher than SS-Net, and it can be seen that our GCN can play a huge role in shop-floor production action recognition tasks.

(2) In terms of temporal positioning for TAD-Net based on TCN, its SL-Score is significantly higher than other mainstream models, and it can be seen that its TCN module based on human skeleton joints sequence flow has a great ability to extract an action’s temporal features. In shop-floor production action temporal positioning tasks, our TCN exceeds LSTM, which is good at sequence extraction, and SS-Net, which is based on rich features video data.

(3) In terms of running speed in the experiment for real-time shop-floor data flow, SS-Net’s running speed is obviously lower than other models based on skeleton joints data because of the huge computing power requirements of real-time video flow data. Due to LSTM itself limiting the computational
efficiency of real-time sequence flow\textsuperscript{12}, the running speed of STA-LSTM and JCR-RNN models based on LSTM is also lower than TAD-Net based on TCN.

Above all, the TAD-Net model based on TCN and GCN has higher recognition accuracy, temporal positioning score and faster detection speed for real-time shop-floor production action. Therefore, our TAD-Net can well meet the digital twin shop-floor’s requirements.

Conclusions

Action detection is one of the main tasks of human management and control in a digital twin shop-floor, and it is also the main difficulty and bottleneck associated with human-computer interaction technology\textsuperscript{13,4}. In order to solve the real-time problem of shop-floor production action detection, we proposed a real-time detection approach. This approach took continuous skeleton joints sequence data as the input, enhanced JCR-RNN’s action temporal positioning module based on TCN, and restructured its recognition and regression module based on GCN. We built our Temporal Action Detection Network (TAD-Net), and realized the real-time detection of the shop-floor production actions. This TAD-Net has the following advantages:

(1) High parallelism. When inputting a time sequence data, TAD-Net’s timing extraction module based on TCN can process several sequence data in parallel, instead of calculating them in order like in Recurrent Neural Network (RNN), such as LSTM\textsuperscript{12}.

(2) Flexible receptive field. The receptive field of timing extraction module is determined by layers of TCN blocks, two-dimensional convolution kernel size and dilated convolution parameters. We can flexibly configure TAD-Net’s receptive field according to different task requirements, and thus get higher temporal positioning accuracy.

(3) Stable gradient. The residual and skip connection structures are extensively used in TAD-Net, and can effectively avoid the gradient disappearance and explosion problem\textsuperscript{19}, thus making TAD-Net’s training simpler and faster.

(4) Lower memory usage. The RNN needs to retain the information for each time period, so it takes up large memory during processing. Comparatively, the convolution kernel of TCN blocks is parameter-sharing, and its memory consumption is lower\textsuperscript{12}.

(5) Higher recognition accuracy. The GCN-based action recognition network can retain the inherent connection information of human skeleton joints in convolution operation, has a stronger ability to extract human limbs features\textsuperscript{5}, and its recognition is more accurate.

Therefore, this approach can better meet actual application requirements and has important research value and practical significance for standardizing shop-floor production processes, reducing production security risks, and contributing to the understanding of real-time production actions.

Of course, our approach still has many shortcomings, so here is a summary and outlook to point out the direction for future work:

(1) Optimized for body covering
Due to the fact that the human body is often covered by machinery in an actual shop-floor, it is necessary to consider the correction and complement processing of disorder or missing skeleton joints data due to the joints being covered.

(2) Interaction with the scene
Our approach is based on the human skeleton joints, and only considered the human limbs’ posture in action recognition tasks. The interaction between the human and the scene, including object information on what the human is holding, is ignored. In future work, we will add object interaction information into TAD-Net to improve action detection performance.

(3) Action multi-label recognition
The output result of real-time production action recognition designed in this paper has only one category, but people may perform multiple actions at same time, such as relying on products to smoke, therefore, action multi-label recognition tasks need to be considered in our future research.

Data availability

Underlying data

The OAD dataset\textsuperscript{10} was captured using the Kinect (V2) sensor, which collects color images, depth images and human skeleton joints synchronously. The dataset includes 59 long sequences and 10 actions, and its data can be viewed by using all versions Notepad++. The OAD dataset is publicly available here: https://www.icst.pku.edu.cn/struct/Projects/OAD.html.

This dataset was not generated nor is it owned by the authors of this article; the listed owners are Yanghao Li, Cuiling Lan, Junliang Xing, Wenzhu Zeng, Chunfeng Yuan, Jiaying Liu. Therefore, neither the authors nor Digital Twin are responsible for the content of this dataset and cannot provide information about data collection. As this dataset contains potentially identifying images/information, caution is advised when using this dataset in future research.


This project contains the following underlying data:

- NJUST3D_dataset.rar (continuous shop-floor production action skeleton joints sequence data collected in the actual shop-floor environment by our team, captured using the Kinect (V2) sensor. The file includes 13 ultra-long-frame sequences and 18 production actions. The data can be viewed by using all versions Notepad++). In this folder, the ‘action.txt’ file is the action sequence, the ‘label_class.txt’ is the label of the action class, and the ‘label_frame.txt’ is the label of the temporal frame).
- Test_raw_data_English.rar (raw data of validation experiment, including models predicting results and real-time action detection time cost. The data can be viewed by using all versions Notepad++). In the ‘predict_results’ folder, each line of the TXT file represents the predicted value output by the model in this frame. The ‘running_speed_test’ folder includes the sequential scripts, the raw skeleton joints sequence data, and model test data. Each line of model test TXT file represents the time cost of model for this action).

- ConfuseMatrix1.csv (data for drawing the confusion matrix).

- Table.xlsx (data for drawing curves of recognition accuracy, SL-Score and running speed).

- Experiment_record.docx (data of the experimental verification, including the experiment data of recognition accuracy, SL-Score and running speed).

Data are available under the terms of the Creative Commons Zero “No rights reserved” data waiver (CC0 1.0 Public domain dedication).


References


http://doi.org/10.23728/b2share.47cfd2b9a24364a16b3481785e a06551

This project contains the following underlying data:
- Gendata.rar (custom code for human skeleton joints preprocessing, including translation, rotation, and normalization).

Codes and scripts are available under the terms of the BSD-3-Clause.

Software availability
Source code available from: https://github.com/1538023886/TAD-Net/tree/main

Archived source code at time of publication: https://doi.org/10.5281/zenodo.54841062

License: BSD-3-Clause

Acknowledgements
Thanks for the assistance of Pei Zhang, Chentao Yang, Yuhang Li and other students from Nanjing University of Science and Technology in the process of data collection.


