Paper:

A Two-stream Bi-directional LSTM Network for Automatic Counting and Localizing Repetitive Actions

Longsheng Wei*[1,2,3], Yuyang Ye[1,2,3], Xuefu Yu[1,2,3] and Dapeng Luo[4]

[1] School of Automation, China University of Geosciences, Wuhan, China
[2] Key Laboratory of Geological Survey and Evaluation of Ministry of Education, Wuhan, China
[3] Hubei key Laboratory of Advanced Control and Intelligent Automation for Complex Systems, Wuhan, China
[4] School of Mechanical Engineering and Electronic Information, China University of Geosciences, Wuhan, China
*E-mail: weilongsheng@cug.edu.cn

Abstract. Latent information includes geological information and drilling progress can be inferred by how long it takes to drill pipes and the number of pipes which are used to drill in a coal mine well site. Since deep learning has made great achievements in the field of computer vision recent years. This work aims at developing a deep learning model for automatic localization when the drilling process happened and how many times it has happened through surveillance videos. We use frames as appearance stream and pixel trajectories of a frame as motion stream in which the displacement values corresponding to a moving scene point are at the same spatial position across several frames. The two-stream CNN is followed by a bi-directional Long Short-Term Memory (LSTM) layer. Instead of predict a video-level classification, our model output a frame-level classification sequence for temporal localization and counting drilling process. We train and test our model using our own dataset which straightly come from coal mine well site which is relatively small compared to common temporal action detection dataset. Our model performance well at test and opens the possibility to validate an automatic methodology for the automatic localization and counting of repetitive actions in industrial environment based on machine-learning algorithm.

Keywords: Temporal action detection, Repetitive actions counting, Deep learning

1. INTRODUCTION

In the coal mine well drill pipe working scene, latent information, which includes geological information and drilling progress, can be known by the duration of drilling pipes and the number of pipes which are used to drill. Traditionally this task has been done manually, which suffered from poor efficiency and human error. Nowadays, with the development of digital cameras and Internet, video becomes easier to capture and distribute. Computer vision based methods, which has been used for object recognition [1,2], are increasingly used on all kinds of jobsites for more effectively collecting and analyzing latent information [3,4].

Our work aims to answer two questions:(1) how many pipes have been drilled in or drilled out;(2) when the pipes were drilled in or drilled out. Reference [3,5] adopted the Bag-of-Feaures pipeline to recognize workers and equipment activities. Their models are based on pre-segmented video clips each contains a single action instance instead of repetitive actions or sequential actions. These methods focus on action recognition for video clips instead of localization repetitive actions in untrimmed videos. What's more, temporal information of action transitions, which is valuable for process flow analysis, is lost due to pre-segmented video clips. Reference [6] are to treat action segmentation and recognition separately. Segmentation methods, e.g. sliding windows, usually based on low-level cues, which leads to high computation burden and imprecise temporal localization [7,8].

Recent years, deep learning methods have made remarkable achievement in temporal action detection [9]. Reference [10,11] showed that results improved when optical flow was fed as motion information. The use of long shot-term memory (LSTM) network improves the training ability of temporal deep learning models [11,12]. However, there exists a considerable gap between common dataset and real world scenarios. There are clear intervals between a large portion of actions in common dataset. Therefore, label of common dataset is coarse-grained. Instead of well-controlled environment, surveillance video from coal mine well are often captured in inappropriate camara angle with object occlusion. And there is little interval between actions because workers try to achieve high efficiency. We label workers' actions in a finer way to train our model.

This research tackles the problem of workers' action detection in continuous, untrimmed video streams from coal mine well. This is a difficult task that there is nearly no interval between actions and some action instances are incomplete. To address this problem, we split the action into three more fine-grained parts. We seek to achieve our goal by frame-wise classification. Inspired by [11,12], We develop a two-stream bi-directional long short-term memory neural network to automatically detect workers' actions. Our network starts with a convolu-

tional neural network (CNN) that has two streams: pixel trajectories as motion and frames as appearance. After feature fusion, the two-stream network output is a temporal feature sequence. We choose long shot-term memory (LSTM) network to model temporal dynamics within and between actions. Compare to traditional recurrent neural networks (RNN), input gates and forget gates allow LSTM to control its memory cell which is more conducive to model temporal sequence. In this application, omitting action instances has a huge impact on the number of pipes that are used. Inspired by [13], we add a confidence module to regress a possibility that a frame contains drilling progress. In training, our loss function consists of two loss: classification loss and confidence loss which is computed by the distance between predictions and groundtruth as confidence loss. At inference time, our network output frame-wise classification sequence. After a simple de-noising method, it is easy to locate duration and count how many pipes are used.

2. RELATED WORK

Detecting human actions is crucial to video understanding. This task has long been an important research topic in computer vision due to the explosive growth of video data and its broad prospect. The orthodoxy of action recognition and detection was to compute hand-crafted features such as Histogram of Oriented Gradients (HOG), Motion Boundary Histogram (MBH), and Histogram of Optical Flow (HOF) along improved dense trajectories [14], create a Fisher vector for each video clip, then using support vector machines to perform classification. Reference [5] showed that deep features which were computed by convolutional neural networks whose inputs were images and stacked optical flow along trajectories outperform hand-crafted features.

Temporal action localization, like object detection, falls under the umbrella of visual detection problems. Inspired by R-CNN series methods which have made remarkable achievements in object detection, "proposal + classification" two-stage pipeline become mainstream in temporal action detection. Reference [7] splits video into clips in different size and applies a binary classifier to rank the sliding windows. [8] use C3D network extract features to create a temporal feature sequence. Then they adopt temporal regression in additional to binary classification to adjust the boundary of sliding windows. Reference [15] applies binary classification on frame-level which represent the possibility that this frame contains an action instance. Then their model adaptively selects sliding windows to fill the omitted ones in proposals. "proposal + classification" pipeline based methods are trained on common dataset which is more variety but less repetition. Reference [13] aims to address the online action detection in such a way that it can predict the action at each time slot efficiently without requiring a sliding window design. They use the regression design to determine the start/end points learned in a supervised manner during the training, enabling the localization being more accurate. In our coal mine well situation, next action instance starts right after the last one. Confidence scores of our videos would be high all the time. We turn this task into a frame-level classification problem. Instead of compute the possibility that this frame contains an action instance, we add a confidence module to output the possibility that this frame is an action instance of drilling.

The use of "CNN + LSTM" to examine spatial and temporal information is an area that has received a considerable amount of interest. In [10], a two-stream network was proposed in which video frames and stacked optical flow were fed to a deep convolution neural network for action recognition. Reference [11] add two person-centric streams to their network. Their ablation experiments demonstrated that each unidirectional LSTM provides a significant boost because it provides more temporal context than a unidirectional LSTM. We develop a two-stream network to extract appearance and motion feature instead of multi-stream proposed by [11] due to the inappropriate camera angle. It is often the case that workers are not fully captured by the camera or workers block the machine. Object detector will fail to detect workers' location.

3. APPROACH

Our framework is shown in Fig. 1. First, we train two independent convolutional neural networks, each based on the VGG16 architecture. As shown in Fig. 1, two of the networks (one each for appearance and motion) are trained on chunks of full-frame video, so that the spatial context of actions being performed is preserved. To capture short-term temporal information, we use pixel trajectories proposed in [11], in which each moving scene point is in positional is consistent across several frames. The calculation of Pixel trajectories is shown in the Fig. 2. More divisible motion information is captured due to this alignment. After two-stream network is a fully-connected projection layer on top of all four fc7 layer outputs, to create a joint representation for these two independent streams. This two-stream network (TSN) is shown in Fig. 1. We do not use objection detection method to track workers as two extra stream which proposed in [11] due to the fact that workers are not fully captured in a fair amount of time. Machine in our video is also hard to track because it is often blocked by workers.

To obtain frame-wise classification, our method uses LSTMs to Associate inter-frame information and learn long-term temporal context of actions in a data-driven fashion. As illustrated in Fig.1, fusion features provide by TSN are fed into an LSTM network running in two directions to better preserve temporal context. For classification, we use two fully connected (FC) layers on top of each directional LSTM's hidden states, followed by a softmax layer, to obtain an intermediate score corresponding to each action. Finally, the scores for the two LSTMs are averaged to get action-specific scores.

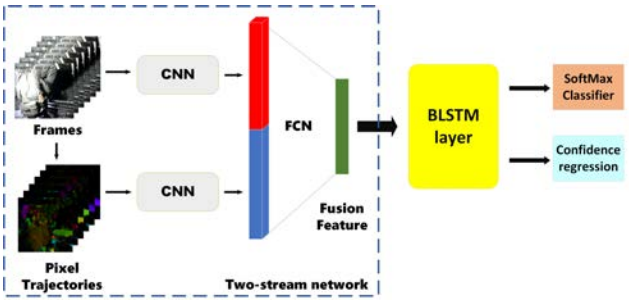

Fig. 1. Architecture of our two-stream Bi-directional LSTM framework.

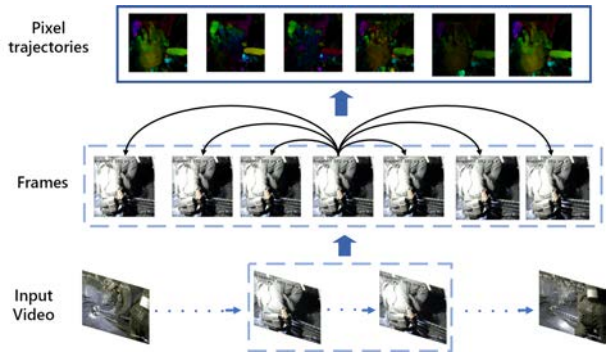

Fig. 2. Pixel trajectories

There are multiple components in an action detection pipeline that are vital for achieving good performance. In this task, some action instances are incomplete. We need a model that omits action instance as little as possible and output the number of action instances. We split the action into three different stages (starting, course and ending) and add a confidence module for fewer omissions. We use two more FC layers on top of each directional LSTM's hidden states for confidence regression. Our confidence measures the possibility of the current frame to be with in a drilling progress. To better localize the start point, we use a Gaussian-like curve to describe the confidences, which is highest at the center of starting stage. Confidence loss is computed by the distance between prediction and groundtruth of confidence sequence.

3.1. Two-stream network

Using stacking optical flow as short-term context information in deep network has been a standard practice in the reference [5,10]. However, motion vectors corresponding to specific moving points in the scene change their pixel positions from one frame to the next in a stacked optical flow. Therefore, the convolutional neural network needs to learn the spatial motion of optical flow to classify the actions. Complete motion information can be learned at a higher level via the network, but this requires more parameters and data to be learned. We follow an alternate representation for motion in a sequence of frames which is to calculate the flow from the center frame t to the first 3 frames and the last 3 frames. In the pixel trajectory, the position of the optical flow image is fixed, and only the intensity changes. Therefore, it is easier for the network to learn the temporal changes of each pixel than from the stacked flow field.

VGG16 configurations are quite different from the ones used in the top-performing entries of the ILSVRC-2012 and ILSVRC-2013 competitions. Rather than using relatively large receptive fields in the first conv. layers e.g. 11×11 with stride 4, or 7×7 with stride 2. VGG16 uses very small 3×3 convolution kernels to perform convolution with the input of each pixel in stride of 1. This reduces the number of parameters: assuming that the input and output of the three-layer 3×3 convolution stack have C channels, the parameter size of the stack is set to $27C^2$;While performing a 7×7 convolution layer will require the $49C^2$ parameter, that is an increase of 81%. Therefore, 3×3 convolution kernels can be regarded as imposing regularization on 7×7 convolutions, forcing them to be decomposed by 3×3 convolution kernels.

We use two independent pretrained VGG16 ConvNet as backbone feature extractor. The output of last convolution layer in VGG16 is considered as the output of this stream. Output from both streams is interrelated after a projection layer which is consisted of two cascaded FC layer. Person tracking streams are abandon due to the fact that camera at drilling site often fail to capture the whole worker.

3.2. Bi-directional LSTM

The LSTM neural network is used to process the sequence input. By mapping the input sequence to the hidden state sequence, RNNs can learn complex temporal dynamics. However, RNNs often suffer from vanishing and exploding gradient when learning long-term dynamics. LSTMs provide a solution by merging memory cells which consist of four gates, input gate, forget gate, output gate and memory gate. The weight update equations for an LSTM cell are as follows:

$$
\begin{aligned}
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \\
g_t &= \tanh(W_{xc}x_t + W_{ho}h_{t-1} + b_c) \\
c_t &= f_t c_{t-1} + i_t g_t \\
h_t &= o_t \tanh(c_t)
\end{aligned}
\quad \ldots \ldots \ldots \ldots \ldots \ldots \ldots \quad (1)
$$

Where σ is a sigmoid function, tanh is the hyperbolic tangent function. i_t, f_t, o_t and c_t are the input gate, forget gate, output gate, and memory cell activation vectors, respectively. The forget gate f_t computed by current input and last hidden state decides which information should be cleared from the memory cell c_t. The input gate i_t and the tanh layer g_t that also computed by current input and last hidden state decide which new information should be incorporated into the memory. More specifically, i_t decides which value will be updated. The tanh layer g_t is a vector generator of new candidate values formed by the tanh

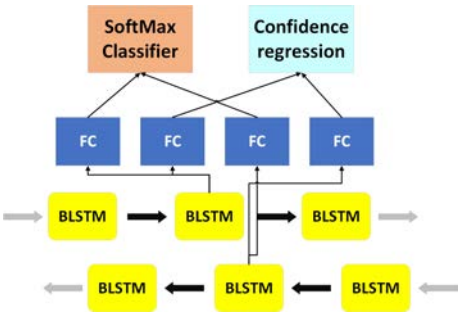

Fig. 3. Classification and regression module

layer. The memory cell c_t is updated based on the output of the forget gate f_t, input gate i_t and the new candidate values g_t.The output gate o_t controls which information in the memory cell should be used as a representation for the hidden state. Finally, the hidden state computed by the memory cell state and the output gate is represented as the final output of a LSTM cell.

As illustrate in Fig. 1, Fusion feature, output of the projection layer, fed into an LSTM network running in two directions. On top of the hidden state of each directed LSTM, we use two cascaded FC layers, followed by a softmax layer to obtain intermediate scores. Finally, the scores of the two LSTMs are averaged to obtain action-specific scores as illustrate in Fig. 3.

Action recognition models based on LSTMs usually predict at frame-level then output a single label for the entire video. While in action detection we need to assign a label for each frame to predict duration. It becomes a problem when correct predictions are not contiguous. To tackle this problem, we apply a simple de-noising method. We split the whole video into 10-frame segments and replace all predictions in the segment with the most categories. With the frame-level prediction, we know what happened at what time. We take segments of drilling process and get rid of those whose interval from the last time is less than 200 frames. The final counting and duration output are obtained by segments that are left. Classification loss is formulated as:

$$L_c(V) = -\frac{1}{N}\sum_{t=0}^{N-1}[\sum_{k=0}^{M} z_{t,k} ln P(y_{t,k}|v_0,...,v_t)] \quad (2)$$

Where $z_{(t,k)}$ corresponds to the groundtruth label of frame v_t for class k. $z_{(t,k)} = 1$ means the groundtruth of frame v_t is the kth class. $y_{(t,k)}$ is the results of frame v_t predicted by the model.

3.3. Confidence Regression

The actions in each video are repeated several times. Inspired by [13], we compute confidences, using FCN just after LSTMs, to predict current frame contains a drilling action. We use a Gaussian-like curve to describe the groundtruth of confidences, which centralizes at the center of drilling action instance as illustrated in Fig. 4. The confidence c_t of the frame v_t is defined as:

$$c_t = -e^{-(t-sj)^2/\sigma^2} \quad . . . . . (3)$$

Where s_j is the center of the nearest (along time) drilling action instances to the frame v_t. σ is the parameter which controls the shape of the confidence curve. The confidence value is 1 when $t = s_j$. For the Gaussian-like curve, a lower confidence value suggests the current frame has larger distance from the center. By contrast, the peak point indicates the center of a drilling action instance. Similar to classification, we use a fully connected network on top of the hidden state of each directed LSTM for regression.

The final loss function of our model is as follows:

$$\begin{aligned} L(V) &= L_c(V) + \lambda L_r(V) \\ &= -\frac{1}{N}\sum_{t=0}^{N-1}[\sum_{k=0}^{M} z_{t,k} \ln P(y_{t,k}|v_{0,...,}v_t) \\ &\quad + \lambda \cdot l(c_t, p_t)] \end{aligned}$$

. . . . . (4)

Where p_t are the predicted confidence values of current frame, λ is the weight for the confidence regression task, l is the regression loss function, which is defined as $l(x,y) = (x-y)^2$. In the training, the overall loss is a summarization of the loss from each frame v_t, where $0 \le t < N$. For a frame v_t, its loss consists of the classification loss represented by the cross-entropy for the $M+1$ classes and the regression loss for identifying the start and end of the nearest action.

4. EXPERIMENT

4.1. Dataset

This experiment based on four surveillance videos from coal mine well site. Videos fall into two events: the iron pipes are (1) drilled in and (2) drilled out. There is nearly no interval between them because workers try to achieve high efficiency.It is not reasonable to label just two categories for boundaries between instance would be unknown. Inspired by [16], we argue that tackling this challenge requires the capability of temporal structure analysis, or in other words, the ability to identify different stages e.g. starting, course, and ending, which together decide the completeness of an actions instance. The videos are labeled frame-wise from 7 classes (six in Fig. 4 and one for background). Video we used is relatively small compared to common action detection dataset and the total length of these videos is 33 minutes. We dissociate 28 minutes (More than 40000 frames) for both training and verification. The remaining 5 minutes (about 7000 frames) serve as a test dataset.

4.2. Implementation details

We build our model using Pytorch. Pixel trajectories are computed by each frame and its 6 neighboring frames as optical flow stream as in [5]. We use epic flow to compute optical flow, as it gives reliable flow even for large movements. Frame sequence and pixel trajectories are resized to 256×256 then fed into VGG16. We project the

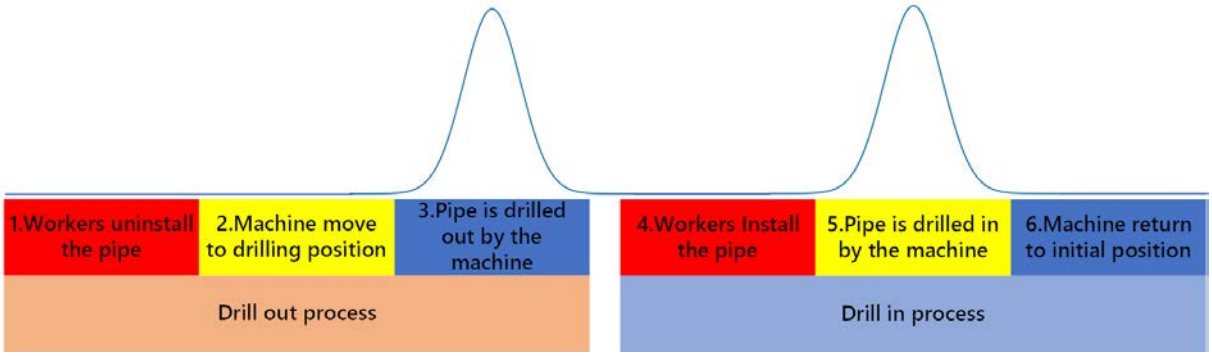

Fig. 4. Categories of each events (with their serial number) and confidence groundtruth

output of two-stream network using a FC layer to a 200-dimensional vector as fusion feature. Then, fusion feature is given to Bi-LSTM layers with 60 hidden units and 20 batchsize each. For classification, forward and backward hidden units are projected to two 7-dimensional vectors through two FC layers of two layers.

These 7-dimensional vectors are averaged and given to a softmax classifier. The softmax predictions of both LSTM networks are averaged to get the action scores for each class. We add a confidence module in training to achieve the goal of less missing action instance. Similar to classification, two FC layers of two layers are used to project hidden units to two scalars. Final confidence equals the average of two scalars. In addition, we used a smoothing method to avoid the interference of individual errors. We extract 10 predictions of 10 frames at a time and replace these 10 predictions with the one that appears most frequently in the 10 frames. The final model shows more robust and perform batter with this smoothing method. We set λ to 1 and use Adam optimizer with a learning rate of 0.0001 for training. We split the whole video into 10- frame segments and replace all predictions in the segment with the most categories as a simple de-noising method. With the frame-level prediction, we know what happened at what time. We take segments of drilling process and get rid of those whose interval from the last time is less than 200 frames. The final counting and duration output are obtained by segments that are left.

4.3. Result

First, we compare our prediction with groundtruth and visualize the result of the comparison in Fig. 5. The horizontal axis is time and different color represent different actions we labeled. Except for the last action instance in which workers uninstall the pipe, our model has shown a lot of qualities predicting durations. For the last action instance, our model incorrectly predicts two segments in this period as other categories. We observe that the last action instance lasts longer than all the previous ones, including the same action in the training set. We replay the video and find out that there is a pause in this action instance. What's more, in the training set, there is an action instance with irregular action. During a drilling out

Fig. 5. Visualization of prediction and groundtruth.

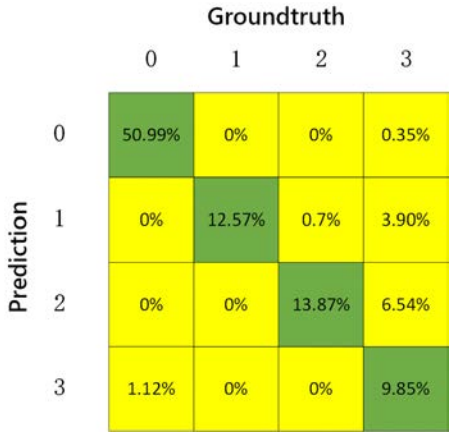

Fig. 6. Frame-level classification confusion matrix.

process, a worker stopped the machine and adjust the position of pipe. To continue this process, the machine has to go back a little so the worker can reconnect the pipe with the machine. We label "Machine move to drilling position" when the machine go back when caused the incorrect prediction we assumed. Note that background is well predicted while the length of background in training set is only 1% of the length of the entire video.

Performance of each category is reported in the confusion matrix in Fig. 6. Green cells indicate true positives, which are when classifier correctly estimates the technical action, while yellow cells indicate false positives and false negatives. Pipes in test video are only drilled out so there are 4 categories in the confusion matrix. It is worth highlighting that the worst performance was obtained for "drilling out" action. In general, our model achieved 91.99% precision at frame level in the test set as illustrate in table 1 where "Precision" is the precision for all frames and "precision*" is the precision for frames without background. CR stands for Confidence Regression.To prevent

Table 1. Precision of each architecture.

Method	*Precision*	*Precision**
Two-stream Bi-LSTM + CR	91.99%	85.62%
Two-stream Bi-LSTM	87.28%	75.79%

Table 2. Performance on the category labeled as 3 in Fig. 4.

Method	*Precision*	*Recall*	*mIoU*
Two-stream Bi-LSTM + CR	92.82%	70.53%	68.16%
Two-stream Bi-LSTM	89.75%	47.72%	52.22%

background, which is a huge proportion of the total video, from affecting the results. Our model still achieved about 85.62% without background part. Our model output correct number of times thanks to the extra criterion we add. We verify the validity of confidence module in ablation experiment by trained a same two-stream Bi-LSTM model without confidence module. In order to quantitative analysis the performance of two architectures, we evaluate mean intersection over union (mIoU), precision and recall for "drilling" category specifically. In Table 2 we compare the performance of our "CNN+LSTM" network with or without confidence module. Including a confidence module improves performance at each metric which means our model learn the representation of drilling process more explicitly by confidence module. The significant increase in recall means that our models can make fewer omissions for drilling process. Another metric, mIoU, is used to measure the degree of alignment between predicted duration and groundtruth. The performance drops by 15.94% if we remove confidence module. We report frame-level precision in test set with or without background of confidence-removed model which also output a correct counting with extra criterion.

5. CONCLUSION

To automatic analyzing drilling process in videos from coal mine well scene, we labeled our own dataset. We used a two-stream bi-directional LSTM network to detect workers' action. Instead of classification in video-level, our model generates frame-level prediction for localization repetitive action instances in untrimmed video which is more challenging. We developed a confidence module for learning more explicit representations of drilling process. We showed that for this task, performance is improved by our confidence module.

In the future, we are going to refine our model from the following two aspects. For one thing, we will investigate replay Bi-LSTM with attention mechanism which processes sequences with less computation. Moreover, the classification of the frames will consider different contexts according to different positions when using LSTM. For example, 10 frames will consider 10 frames before and after, while 20 frames can only consider the last 20 frames. For another, workers may block the lens in videos for a short time, which makes it difficult to classify the lens. If attention can be used to calculate the similarity and assign small weights to the blocking part, the detection effect may be improved to some extent.

Acknowledgements

This work was supported by the Joint Foundation of China Aerospace Science and Industry for Equipment Pre Research 2020, the Opening Fund of Key Laboratory of Geological Survey and Evaluation of Ministry of Education (Grant No. GLAB2020 ZR06) and the Fundamental Research Funds for the Central Universities, and the National Natural Science Foundation of China under contracts 61603357.

References:

[1] Ren S , He K , Girshick R , et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2017, 39(6):1137-1149.

[2] Redmon J, Divvala S K, Girshick R, et al. You Only Look Once: Unified, Real-Time Object Detection[C]. Computer Vision and Pattern Recognition, 2016: 779-788.

[3] Ding L , Fang W , Luo H , et al. A deep hybrid learning model to detect unsafe behavior: Integrating convolution neural networks and long short-term memory[J]. Automation in Construction, 2018, 86:118-124.

[4] Gong J, Caldas C H, Gordon C, et al. Learning and classifying actions of construction workers and equipment using Bag-of-Video-Feature-Words and Bayesian network models[J]. Advanced Engineering Informatics, 2011, 25(4): 771-782.

[5] Wang L, Qiao Y, Tang X, et al. Action recognition with trajectory-pooled deep-convolutional descriptors[C]. computer vision and pattern recognition, 2015: 4305-4314.

[6] Weinland D, Ronfard R, Boyer E, et al. A survey of vision-based methods for action representation, segmentation and recognition[J]. Computer Vision and Image Understanding, 2011, 115(2): 224-241.

[7] Shou Z, Wang D, Chang S, et al. Temporal Action Localization in Untrimmed Videos via Multi-stage CNNs[C]. Computer Vision and Pattern Recognition, 2016: 1049-1058.

[8] Gao J, Yang Z, Sun C, et al. TURN TAP: Temporal Unit Regression Network for Temporal Action Proposals[C]. International Conference on Computer Vision, 2017: 3648-3656.

[9] Lin T, Zhao X, Su H, et al. BSN: Boundary Sensitive Network for Temporal Action Proposal Generation[C]. European Conference on Computer Vision, 2018: 3-21.

[10] Simonyan K, Zisserman A. Two-Stream Convolutional Networks for Action Recognition in Videos[C]. Neural Information Processing Systems, 2014: 568-576.

[11] Singh B, Marks T K, Jones M, et al. A Multi-stream Bi-directional Recurrent Neural Network for Fine-Grained Action Detection[C]. Computer Vision and Pattern Recognition, 2016: 1961-1970.

[12] Sun C, Shetty S, Sukthankar R, et al. Temporal Localization of Fine-Grained Actions in Videos by Domain Transfer from Web Images[C]. acm multimedia, 2015: 371-380.

[13] Li Y, Lan C, Xing J, et al. Online Human Action Detection Using Joint Classification-Regression Recurrent Neural Networks[C]. European Conference on Computer Vision, 2016: 203-220.

[14] Wang H, Schmid C. Action Recognition with Improved Trajectories[C]. International Conference on Computer Vision, 2013: 3551-3558.

[15] Gao J, Chen K, Nevatia R, et al. CTAP: Complementary Temporal Action Proposal Generation[C]. european conference on computer vision, 2018: 70-85.

[16] Zhao Y, Xiong Y, Wang L, et al. Temporal Action Detection with Structured Segment Networks[C]. international conference on computer vision, 2017, 128(1): 2933-2942.