electronics

MDPI

Article

BTM: Boundary Trimming Module for Temporal Action Detection

Maher Hamdi [1], Shiping Wen [2] and Yin Yang [3,*]

[1] School of Business, George Washington University , Washington, DC 20052 , USA
[2] Australia AI Institute, University of Technology Sydney, Ultimo 2007, Australia
[3] College of Science and Engineering, Hamad Bin Khalifa University (HBKU), Doha 5825 , Qatar
* Correspondence: yyang@hbku.edu.qa

Abstract: Temporal action detection (TAD) aims to recognize actions as well as their corresponding time spans from an input video. While techniques exist that accurately recognize actions from manually trimmed videos, current TAD solutions often struggle to identify the precise temporal boundaries of each action, which are required in many real applications. This paper addresses this problem with a novel Boundary Trimming Module (BTM), a post-processing method that adjusts the temporal boundaries of the detected actions from existing TAD solutions. Specifically, BTM operates based on the classification of frames in the input video, aiming to detect the action more accurately by adjusting the surrounding frames of the start and end frames of the original detection results. Experimental results on the THUMOS14 benchmark data set demonstrate that the BTM significantly improves the performance of several existing TAD methods. Meanwhile, we establish a new state of the art for temporal action detection through the combination of BTM and the previous best TAD solution.

Keywords: action detection; boundary trimming module; video analytics

Citation: Hamdi, M.; Wen, S.; Yang, Y. BTM: Boundary Trimming Module for Temporal Action Detection. *Electronics* **2022**, *11*, 3520. https://doi.org/10.3390/electronics11213520

Academic Editors: Yohan Ko, Gwanggil Jeon and George K. Adam

Received: 25 August 2022
Accepted: 22 October 2022
Published: 29 October 2022

1. Introduction

Temporal action detection (TAD) aims to identify actions from an input video, as well as the start and end frames for each action. Most existing TAD methods, e.g., [1–7], are based on video classification [8–11], which classifies a (often trimmed) video clip but does not return any temporal information. As an analogy, TAD to video classification can be compared to object detection in still images [12–15] to image classification [7,16]. Similar to the fact that the resolution of images has strong implications in object detection results, the temporal resolution of the video directly influences TAD performance. This motivates the use of video interpolation methods [17,18] to improve the overall results.

The performance of TAD methods depends upon the underlying video classification networks [9–11], especially when pre-trained networks are used to extract features of the video. The current video classification networks can be classified into two categories: one-stream (e.g., C3D [19] and 3D-ResNet [20]) and two-stream (e.g., TSN [11], I3D [21], T3D [22], and P3D [23]). One-stream models only utilize the pixel color (i.e., RGB) information of the input video, whereas two-stream models take as input pixel colors with optical flow information, which can be obtained through a pre-processing step. Usually, two-stream models obtain higher accuracy at the expense of high computational costs to extract optical flows.

Most existing TAD methods on untrimmed video generally start with extracting features of the video using a classification neural network. For instance, Ref. [24] discusses object tracking in video, using a model based on a combination of Yolo v3 and video processing modules. Ref. [25] performs video action recognition using several techniques such as vector autoregression and wavelet transforms. Then, a TAD model takes these extracted features as input to locate and classify the actions. There are also multi-stage

networks in the field of object detection, such as cascading R-CNN [26]. We can also consider adding more links to a TAD model to readjust the prediction results. The reason is that precisely determining the actual start and end of an action is not practical. Instead, we fluctuate around rough critical points. As a result, the results obtained by current TAD models are not very accurate for the boundaries, especially for settings with high Intersection over Union (IoU) requirements. Thus, it is significant and rewarding to adjust the boundary of the detection results. By utilizing the information around the boundary, the critical information of the action and the non-action, as well as the boundary location, can be more accurately recognized. Finally, recently Transformer-based models have much research attraction, e.g., [27–29]; however, none of them addresses the issues described above.

Motivated by the above discussions, this paper presents a novel Boundary Trimming Module (BTM) for boundary pruning and adjustment of TAD methods. This module can be applied in combination with most existing TAD models. Experimental results using THUMOS14 [30], a popular benchmark data set, demonstrate that the BTM significantly improves the overall performance of TAD methods, especially under high IoU requirements.

2. Related Work

Action recognition. In video analytics, action recognition (e.g., [9–11]) aims to classify video clips based on the actions they present. Usually, each such video clip contains one action, which is (often manually) clipped from a longer, raw video stream that may contain multiple actions. Existing solutions for action recognition include one-stream and two-stream methods. The former are usually faster since they only utilize the pixel color information in the input video without complex video pre-processing. Examples of such methods include C3D mentioned in the Introduction, as well as 3D-ResNet [20], which is based on ResNet [31], and achives much improved prediction accuracy. On the other hand, two-stream methods typically derive motion information in the form of optical flow information using motion displacements between adjacent frames and use such optical flows in combination with pixel colors to obtain more accurate action recognition results. Notable methods include TSN [11], I3D [21], and P3D [23].

Temporal action detection. Temporal action detection (TAD) [32–40] operates on longer, untrimmed videos and identifies the actions therein, together with their respective start and end frames. Clearly, TAD is a more challenging task compared to action recognition, since (i) the input video may contain a large number of frames not relevant to the detected action, which distract the model, and (ii) the model must also report temporal boundaries of each detected action.

Earlier approaches to TAD apply a sliding window over the time dimension on the input video and classifies the video segment inside the window. Since actions may have different lengths, such a method must also try different window sizes, which is highly inefficient. Meanwhile, since it is infeasible to try every possible window size, the reported locations of the start/end frames (i.e., the sliding window boundaries) are usually imprecise. Later, RNN [41] was also used in TAD models as a means to capture temporal correlation information. The use of RNNs, however, does not address the problem that the detected action boundaries are imprecise.

Shou et al. [19] proposed using a C3D network to classify each frame of video, which leads to improved action positioning compared to sliding-window-based methods. More recent approaches, e.g., SSAD [16] and R-C3D [7], achieve more accurate results based on similar ideas as object detection in still images. In addition, BSN [42] focuses on timing positioning, which joins the border-sensitive network and is modified according to the proposal to obtain more accurate boundary locations. Our proposed approach can work with any existing TAD solution and improves its accuracy by refining the boundaries of the extracted actions.

3. Approach

This section presents Boundary Trimming Module (BTM), a novel post-processing module for video TAD. As mentioned in the Introduction, BTM is applied after another TAD solution to refine the temporal boundaries of the detected actions. Specifically, BTM involves two major steps. The first step applies a novel action estimation network (AEN), a two-level convolutional neural network (ConvNet) that classifies each frame image as either background or part of an action. Note that the AEN is a binary classifier, and it does not tell what the action is when it predicts a frame as a part of an action. In particular, the AEN outputs a score for each frame, which reflects its probability of being part of an action. By splicing the scores of the video summary image for each frame, a continuous action score curve can be obtained. After that, the second step of BTM adjusts the temporal boundaries of actions in the video according to a pruning strategy. In the following, we elaborate on the AEN (in Step 1) and the pruning (in Step 2).

3.1. Action Estimation Network

As described above, for each frame in the input video, the AEN outputs its probability of being part of an action. Figure 1 illustrates the architecture of the proposed AEN. As shown in the figure, AEN involves a video feature extractor. Since the proposed BTM is a post-processing step that applies after an existing TAD solution, we simply use the video feature extractor in the partner TAD method. In our experiments, we use the specific feature extractor used in I3D, described in the Related Work section. Note that the video feature extractor is already trained (i.e., as part of the partner TAD solution) before BTM is applied, and BTM only needs to fine tune it for the purpose of AEN.

For the selection of the feature extraction network, though the object of the motion evaluation network to be processed is each frame image in the video, if the image domain is similar to ResNet [31], GoogleNet or NASNet [43] for feature extraction, only the space of the video frame can be extracted. Such a method is only suitable for target detection and classification in images and is obviously not suitable for action understanding with obvious context information. Therefore, the feature extraction network still needs some understanding of the timing information.

As mentioned earlier, C3D and R-C3D are faster, but their accuracy is slightly lower on a video data set such as UCF-101. The features extracted with them will lead to insufficient understanding of the video and will affect the accuracy of the entire action evaluation network. Because the modules in this paper are used for precise adjustment of timing boundaries, this paper hopes to obtain more effective video features, such as I3D [21], MF-Net [44], TSN [11], and other video understanding models. As mentioned above, the I3D network is the best performing network on the UCF-101 data set, which means that the extracted features are most effective for video understanding. Therefore, this paper also uses an I3D network as the feature extraction network.

Since "action" is time-related, it is a continuous change in posture of the human body over time. In order to more effectively judge whether the current frame is an action instance, it is necessary to supplement the context information when extracting features. In this paper, the input of the feature extraction network is centered on the current frame, supplementing 16 frames before and after the frame as input, performing 3D convolution and pooling, and then obtaining the feature map with dimension 1024 as the feature of the current frame by averaging pooling.

The second step is to use the classifier to score. After the feature extraction is completed, the obtained feature is input to the classifier. In the structure of this section, a two-layer full-connection layer with a channel number of 512 is used as a classifier. Since the problem is essentially a classification problem, a softmax layer is connected after the fully connected layer. The loss function is a cross entropy loss function with a formula of (1), where y_i represents the actual output of the network, and d_i represents the desired output.

$$Loss = -\frac{1}{n}\sum_{i=1}^{n} d_i log(y_i) \quad (1)$$

Since the final output is a probability of actions, it is necessary to limit the output to the interval [0, 1]. Therefore, the final output is normalized using the sigmoid function, as shown in Equation (1), where s_a is the final action score, and o_n is the network output. After that, the action scores of each frame are smoothly connected. Finally, a continuous action score curve is obtained. As shown in Figure 2, the horizontal axis is the frame number, and the vertical axis is the action score.

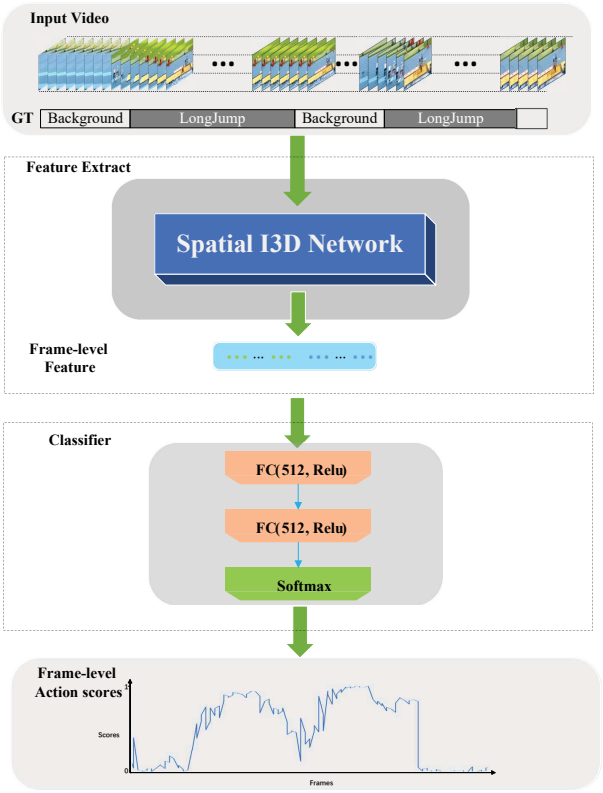

Figure 1. Architecture of the proposed action estimation network.

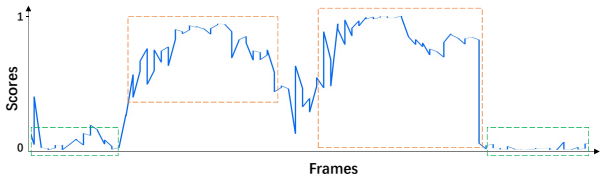

Figure 2. Example video action score curve output by the action estimation network.

3.2. Timing Boundary Trimming Strategy

When analyzing the action score curve of a video, the segment containing the action instance is intuitively the portion of the curve that reaches a continuous high score, as covered by the orange dashed box in Figure 2. The background is the continuous low score part of the curve, as covered by the green dashed box in Figure 2. The timing boundaries of this section are between the orange dotted frame and the green dotted frame. As can be seen from Figure 2, the action scores predicted by the network are still subject to certain disturbances. This section presents a robust and effective strategy to locate the boundaries.

The core idea of the proposed temporal boundary trimming strategy is to match the time-series boundary of the prediction result with the action score curve to obtain the score of the current time-series boundary. The proposed timing boundary pruning technique uses an action threshold α, as follows. If the predicted action probability of a frame is higher than α, the frame is considered to be an action class and vice versa. Using this method,

we obtain a temporal boundary (f_s, f_e) of each action instance, where f_s (resp. f_e) is the start (resp. end) frame of the action in the video. Note that since BTM is a post-processing technique for adjusting temporal action boundaries, it takes as inputs the actions already detected by an existing TAD solution.

Algorithm 1 shows the adjustment of the start frame f_s of the time series as an example, f_s corresponds to the position in the action score curve, and the current start frame f_s can be obtained, that is, the intersection of the red dotted line and the blue in Figure 3. Therefore, the m frames before and after the start frames f_s are included in the reference range, the average action score of the $2m + 1$ frame is calculated, and the result is denoted as s_α. Adopting this multi-frame averaging strategy can reduce the impact of action estimation network misjudgment and enhance the robustness of the trimming module. Then, the absolute value of the difference between the calculated s_α and the action threshold α is $diff$. If the difference $diff$ is greater than the tolerance τ, it is proved that the f_s at this time deviates from the actual motion start time, and it needs to be discussed in two cases. In the first case, when the average action score s_α is less than the threshold α, it means that the action has not yet started. The real action start time is after the f_s time, so we need to make $f_s = f_s + step$, and update f_s to the right to move the $step$ frame to the real action start time. In the second case, i.e., $s_\alpha \geq \alpha$, we update f_s to the left to move $step$ frames, i.e., $f_s = f_s - step$. In this way, f_s is updated cyclically until the difference $diff$ is less than the tolerance τ, and the latest f_s value is returned as the motion start frame of the prediction result.

Algorithm 1 Adjust start frame f_s

Reference range $\leftarrow$ m frames before and after f_s
$s_\alpha \leftarrow$ the average action score of the $2m + 1$ frames in the reference range
$diff \leftarrow |s_\alpha - \alpha|$
while $diff \geq \tau$ **do**
  If $s_\alpha < \alpha$ **then**
    $f_s \leftarrow f_s + step$
  else
    $f_s \leftarrow f_s - step$
  end if
  $diff \leftarrow |s_\alpha - \alpha|$
end while

For the adjustment of the end frame f_e in the prediction result, the adopted strategy is basically the same. The only difference is that when the difference $diff$ is greater than the tolerance τ and the average action score s_α is greater than the action threshold α, the time point at which f_e is located is still in the motion progress time, and the real action end time is after f_e. Let $f_e = f_e + step$, then f_e updates the $step$ frame to the right. When the difference $diff$ is less than the tolerance τ and the average action score s_α is smaller than the action threshold α, the time point action of f_e has ended, and the real action end time is before f_e. Let $f_e = f_e - step$, then f_e updates the $step$ frame to the left.

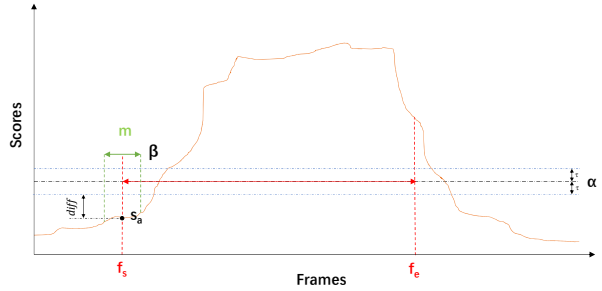

Figure 3. Timing boundary (example values: $f_s = 10$, $f_e = 15$, $m = 3$, $diff$ = 0.05).

4. Experiment and Simulation

4.1. Data Sets and Evaluation Indicators

The data set selected for the experimental simulation in this paper is THUMOS14 [30]. The data set has two tasks: temporal action detection and action classification. The training set is UCF-101 [45], which consists of a video clip containing 101 actions. There are a total of 13320 divided short videos, usually one video contains only one action instance. The validation set and test set include 1010 and 1574 unsplit long videos, respectively, and usually contain multiple action instances. For the temporal action detection task, the data with the start and end time tags that can be used for training and testing has only 21 types of actions, including 200 verification set videos (including 3007 action instances) and 213 test set videos (including 3358 action instances). Each video contains an average of 15 or more actions. Therefore, in the temporal action detection, the verification set of THUMOS14 is used to train the network, and the test set is used to test the performance of the network. It is worth noting that the average length of the video is long, which poses a great challenge to the temporal action detection task. In terms of evaluation indicators, this paper will calculate and analyze the accuracy of the two classifications of the motion assessment network. Then, the time-series boundary refinement module is applied to the existing excellent network, and the average precision mean mAP under each cross-combination IoU is compared. Apparently, the average accuracy mean mAP@0.5 is mostly concerned, when IoU = 0.5.

4.2. Implementation Details

During the classification experiment of the motion evaluation network, it is found that whether the feature information extracted by the RGB stream is not obvious to the overall performance improvement of the model, but it will occupy a large amount of disk I/O time, which leads to a significant increase in the training time of the model. Therefore, in order to save training time, this paper chooses to use only optical flow for feature extraction.

The video is first sampled at a rate of 25 frames per second, and then the TV-L1 algorithm [46] is used to extract the optical stream of the video. Therefore, the feature information of the nth frame in the video can be expressed as $s_n = \{F_n\}_{f_e}^{f_s}$, where F_n represents the optical flow information of the nth frame. When using the I3D network for feature extraction, since the original I3D is 16 frames without repeated downsampling, the motion evaluation of the frame level of this paper cannot be satisfied. Therefore, the sampling mode of the I3D network needs to be modified. After modification, the sampling window length is 16 frames, the initial sampling frame $f_s = n - 7$ of the nth frame feature extraction, and the sampling frame fe is ended. Insufficient start and end are supplemented by blank frames. The sampling diagram is shown in Figure 4. A black rectangle represents a video frame, and a number within a rectangle represents the serial number of the frame. The orange solid rectangle indicates the feature extraction sampling frame of frame 8. The green dotted rectangle indicates the feature extraction sampling frame of the ninth frame of the next frame.

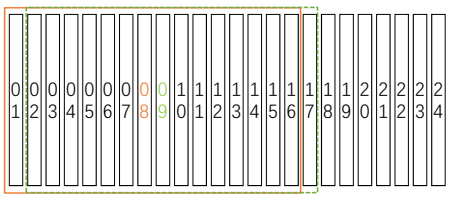

Figure 4. Feature extraction sampling schematic.

Prior to training, the goal of the motion estimation network was to obtain an action score for each frame. In fact, this is a two-category task of background and action. THUMOS14's tag contains 21 categories and timing action start and end time tags, which are not required for motion evaluation network tags, so THUMOS14 tags need to be reprocessed. In this section, the label category of the frame in the video clip with motion tag in THUMOS14

is set to one, and the rest is the background category, and the label is set to zero, so it is converted into a two-class problem. The verification set of THUMOS14 is divided into the training set and the verification set of this experiment, and the division ratio is 3:1.

Furthermore, since the duration of the tagged action instance in the entire video is relatively short, the background category (negative sample) is much more than the action category (positive sample). Such positive and negative sample imbalance problems cannot be ignored. Too many negative samples will affect the direction of parameter update during network training, so that the model tends to predict the sample as a negative sample when judging. In order to solve this problem, this module adopts the method of downsampling the negative samples; that is, the random samples are randomly removed, so that the ratio of the final positive and negative samples is close to 1:1.

In the parameter setting of the timing boundary pruning strategy, after many trials and attempts, the final parameters are set as follows: calculate the average number of frames before and after f_s and f_e, take $m = 2$, update the step size, action threshold $\alpha = 0.5$, endurance Degree $\tau = 0.1$. Furthermore, when the operation time of the THUMOS14 is counted, it is found that the shortest operation time is only 15 frames. Therefore, in order to prevent the error of the timing refinement module from being excessively adjusted, the upper limit of the number of adjustment iterations is set to 15.

4.3. Simulation Results

The results we used to adjust were obtained through a multi-scale detection network that uses different time length feature maps to detect action. The input characteristics of the network are the same as those obtained by I3D extraction in this paper. The network generates proposals by using a multi-scale method. For the second stage, the feature is first input into the bidirectional LSTM so that the network has more recognition of time, and then the RoI pooling is used to generate features for classification and positioning.

For the performance of the motion evaluation network, THUMOS14 is trained with the timing tag verification set, the training accuracy can reach 92.1%, and the test accuracy is about 79.6%. Figure 5 shows the action score curve output by the motion evaluation network test sample, where the solid blue line is the real label, and the red dotted line is the predicted value. Figure 6 shows the accuracy of the classification of each category in the verification set based on the action assessment network score.

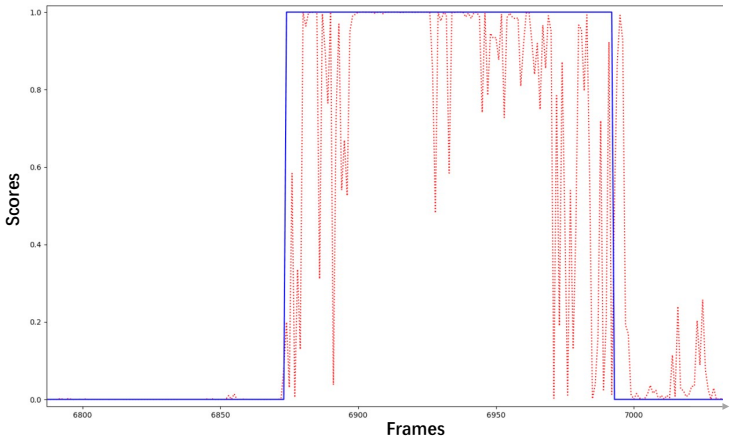

Figure 5. Action estimation network predicts the comparison of the score curve with the label.

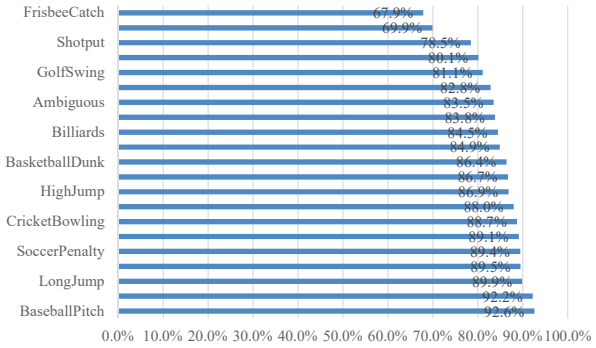

Figure 6. Verify the accuracy of the classification of various actions based on the action estimation network.

The timing boundary module aims at refining the predicted result of the existing sequential motion detection network and can be attributed to the post-processing step of the sequential motion detection network. Therefore, to verify the effectiveness of this module, it is necessary to combine this module with the existing excellent network and compare the changes in network performance before and after the addition of this module. As shown in Table 1, before and after adding the timing boundary trimming module, each network is in the THUMOS14 data set and the average accuracy mean mAP under different IoU.

Table 1. Temporal action detection results on THUMOS'14.

IOU	0.1	0.3	0.5	0.7
Xu et al. [47]	54.5	44.8	28.9	-
BTM + Xu et al.	55.6	45.3	32.4	-
Gao et al. [48]	60.1	50.1	31.0	9.9
BTM + Gao et al.	60.8	52.1	35.6	14.3
Lin et al. [42]	-	53.5	36.9	20.0
BTM + Lin et al.	-	54.3	38.3	21.1
Chao et al. [5]	59.8	53.2	42.8	20.8
BTM + Chao et al.	60.4	53.8	44.6	24.4

It can be found from the results that after adding the time-series boundary refinement module proposed in this section, the key performance index mAP@0.5 is improved, which proves that the proposed module optimizes the time-series positioning of network prediction results so that the start time and deadline of the action are more accurate and have effects on multiple networks. BTM can be added as a general module to the network, providing a new idea for further improvement of network performance.

As shown in Figure 7, our network can identify certain categories of actions very well. In Figure 7, the blue mark is the ground truth, and the number indicates the start and end time of the action. The red mark is the prediction result, which shows not only the start and end times of the forecast but also the confidence of the prediction.

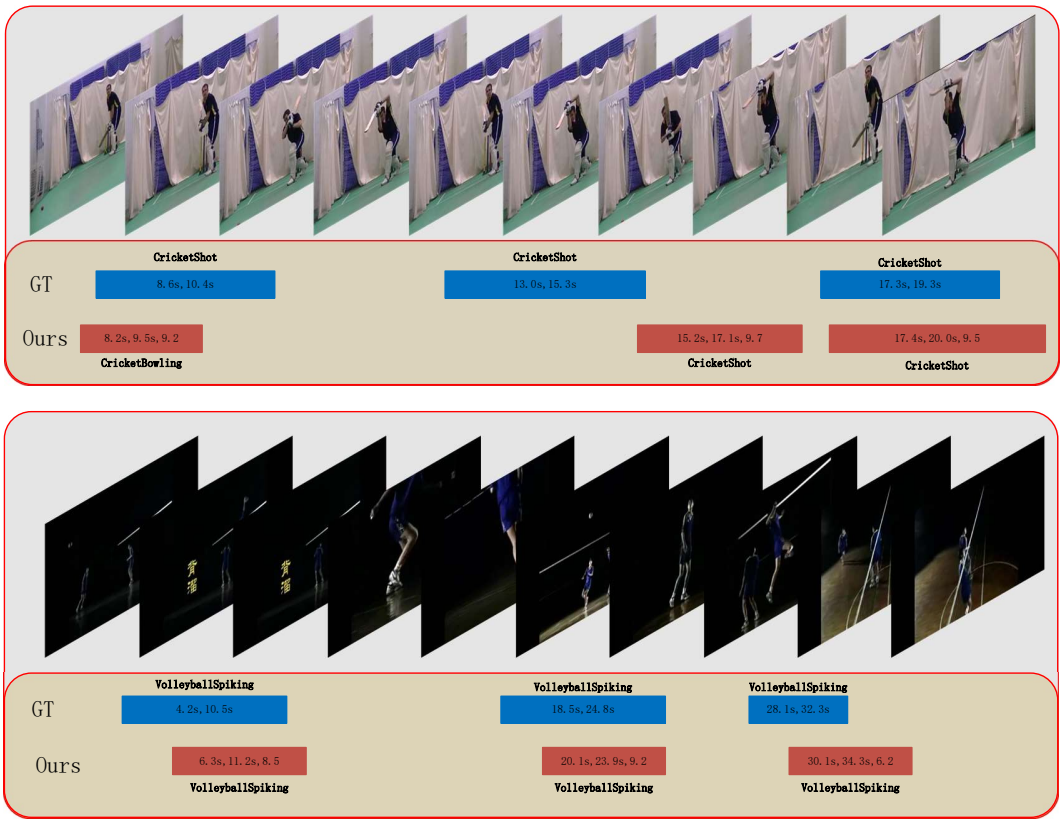

Figure 7. Qualitative examples of the predictions of our network on the THOMOS'14 data set. The blue mark is the ground truth, and the number indicates the start and end time of the action. The red mark is the prediction result, which shows not only the start and end times of the forecast but also the confidence (each scaled by a factor of 10x) of the prediction.

Finally, we mention that adding BTM also leads to increased computational costs. In particular, with our current implementations, we observed up to a 50% increase in inference time in our experimental evaluations. Given the significant accuracy gains of BTM, we would consider this a favorable trade-off for applications that require high temporal detection accuracy.

5. Conclusions

In this paper, a Boundary Trimming Module was proposed to adjust the generated detection results. The module adjusts the temporal boundary according to the classification of the frame level. The network classifies the surrounding frames of the start and end frames of the generated results. By setting a reasonable threshold, the starting and ending time points are adjusted to locate the action more accurately. The THUMOS14 data set was utilized to test the proposed BTM with other networks with greatly improved results, which verifies the versatility of our network. Regarding future work, we plan to further improve the inference efficiency of the BTM module through, e.g., pruning, quantization, and distillation techniques. Another important direction is to explore the adaptation of the proposed BTM module to Transformer-based solutions for temporal action detection.

Author Contributions: Conceptualization, S.W.; Funding acquisition, Y.Y.; Resources, M.H.; Software, M.H.; Supervision, Y.Y.; Writing – original draft, S.W.; Writing – review and editing, Y.Y..

Funding: This publication was made possible by grant RRC02-0826-210048 from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yeung, S.; Russakovsky, O.; Mori, G.; Lei, F.-F. End-to-end learning of action detection from frame glimpses in videos. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2678–2687.
2. Yuan, J.; Ni, B.; Yang, X.; Kassim, A.A. Temporal action localization with pyramid of score distribution features. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 3093–3102.
3. Shou, Z.; Wang, D.; Chang, S.F. Temporal action localization in untrimmed videos via multi-stage cnns. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 1049–1058.
4. Zhu, Y.; Newsam, S. Efficient action detection in untrimmed videos via multi-task learning. In Proceedings of the 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), Santa Rosa, CA, USA, 24–31 March 2017; pp. 197–206.
5. Chao, Y.W.; Vijayanarasimhan, S.; Seybold, B.; Ross, D.A.; Deng, J.; Sukthankar, R. Rethinking the Faster R-CNN Architecture for Temporal Action Localization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 1130–1139.
6. Zhao, Y.; Xiong, Y.; Wang, L.; Wu, Z.; Tang, X.; Lin, D. Temporal action detection with structured segment networks. In Proceedings of the IEEE International Conference on Computer Vision 2017, Venice, Italy, 22–29 October 2017; pp. 2914–2923.
7. Xu, H.; Das, A.; Saenko, K. R-C3D: region convolutional 3d network for temporal activity detection. In Proceedings of the IEEE International Conference on Computer Vision 2017, Venice, Italy, 22–29 October 2017; pp. 5794–5803.
8. Peng, Y.; Zhao, Y.; Zhang, J. Two-stream collaborative learning with spatial-temporal attention for video classification. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *29*, 773–786.
9. Simonyan, K.; Zisserman, A. Two-stream convolutional networks for action recognition in videos. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 568–576.
10. Tran, D.; Bourdev, L.; Fergus, R.; Torresani, L.; Paluri, M. Learning spatiotemporal features with 3d convolutional networks. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 4489–4497.
11. Wang, L.; Xiong, Y.; Wang, Z.; Qiao, Y.; Lin, D.; Tang, X.; Van Gool, L. Temporal segment networks: Towards good practices for deep action recognition. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 20–36.
12. Zhang, X.; Xiong, H.; Lin, W.; Tian, Q. Weak to Strong Detector Learning for Simultaneous Classification and Localization. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *29*, 418–432.
13. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
14. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 91–99.
15. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, 2015; pp. 1440–1448.
16. Lin, T.; Zhao, X.; Shou, Z. Single shot temporal action detection. In Proceedings of the 2017 ACM on Multimedia Conference. ACM, 2017; pp. 988–996.
17. Choi, G.; Heo, P.G.; Park, H.W. Triple-Frame-Based Bi-Directional Motion Estimation for Motion-Compensated Frame Interpolation. *IEEE Transactions on Circuits and Systems for Video Technology* **2018**, *29*, 1251–1258.
18. Wen, S.; Liu, W.; Yang, Y.; Huang, T.; Zeng, Z. Generating realistic videos from keyframes with concatenated GANs. *IEEE Transactions on Circuits and Systems for Video Technology* **2018**, *29*, 2337–2348.
19. Shou, Z.; Chan, J.; Zareian, A.; Miyazawa, K.; Chang, S.F. Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. In Proceedings of the Computer Vision and Pattern Recognition (CVPR) 2017, Honolulu, HI, USA, 21–26 July 2017; pp. 1417–1426.
20. Hara, K.; Kataoka, H.; Satoh, Y. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 6546–6555.
21. Carreira, J.; Zisserman, A. Quo vadis, action recognition? a new model and the kinetics dataset. In Proceedings of the Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on. IEEE, Honolulu, HI, USA, 21–26 July 2017; pp. 4724–4733.
22. Diba, A.; Fayyaz, M.; Sharma, V.; Karami, A.H.; Arzani, M.M.; Yousefzadeh, R.; Van Gool, L. Temporal 3d convnets: New architecture and transfer learning for video classification. *arXiv* **2017**, arXiv:1711.08200.
23. Qiu, Z.; Yao, T.; Mei, T. Learning spatio-temporal representation with pseudo-3d residual networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 5533–5541.
24. Andriyanov, N.; Dementiev, V.; Kondratiev, D. Tracking of Objects in Video Sequences. In *Proceedings of the Intelligent Decision Technologies*; Czarnowski, I.; Howlett, R.J.; Jain, L.C., Eds.; Springer: Singapore, 2021; pp. 253–262.
25. Abdu-Aguye, M.; Gomaa, W. Novel Approaches to Activity Recognition Based on Vector Autoregression and Wavelet Transforms. In Proceedings of the 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), Orlando, FL, USA, 17–20 December 2018; pp. 951–954. https://doi.org/10.1109/ICMLA.2018.00154.
26. Cai, Z.; Vasconcelos, N. Cascade r-cnn: Delving into high quality object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 6154–6162.
27. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *arXiv* **2020**, arXiv:2010.11929.

28. Evdokimova, V.; Petrov, M.; Klyueva, M.; Zybin, E.; Kosianchuk, V.; Mishchenko, I.; Novikov, V.; Selvesiuk, N.; Ershov, E.; Ivliev, N.; et al. Deep learning-based video stream reconstruction in mass-production diffractive optical systems. *Computer Optics* **2021**, *45*, 130–141. https://doi.org/10.18287/2412-6179-CO-834.
29. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. *arXiv* **2021**, arXiv:2103.14030.
30. Jiang, Y.; Liu, J.; Zamir, A.R.; Toderici, G.; Laptev, I.; Shah, M.; Sukthankar, R. THUMOS Challenge: Action Recognition with a Large Number of Classes, 2014. https://www.crcv.ucf.edu/THUMOS14/home.html, accessed in October, 2022.
31. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
32. Gao, J.; Yang, Z.; Sun, C.; Chen, K.; Nevatia, R. Turn Tap: Temporal Unit Regression Network for Temporal Action Proposals, 2017. https://arxiv.org/abs/1703.06189, accessed in October 2022.
33. Dai, X.; Singh, B.; Zhang, G.; Davis, L.S.; Chen, Y.Q. Temporal context network for activity localization in videos. In Proceedings of the Computer Vision (ICCV), 2017 IEEE International Conference, Venice, Italy, 22–29 October 2017; pp. 5727–5736.
34. Hou, R.; Sukthankar, R.; Shah, M. Real-time temporal action localization in untrimmed videos by sub-action discovery. In Proceedings of the BMVC, London, UK, 4–7 September 2017.
35. Buch, S.; Escorcia, V.; Ghanem, B.; Fei-Fei, L.; Niebles, J. End-to-end, single-stream temporal action detection in untrimmed videos. In Proceedings of the British Machine Vision Conference (BMVC), London, UK, 4–7 September 2017.
36. Yuan, Z.H.; Stroud, J.C.; Lu, T.; Deng, J. Temporal Action Localization by Structured Maximal Sums. In Proceedings of the CVPR, Honolulu, HI, USA, 21–26 July 2017.
37. Buch, S.; Escorcia, V.; Shen, C.; Ghanem, B.; Niebles, J.C. Sst: Single-stream temporal action proposals. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6373–6382.
38. Escorcia, V.; Heilbron, F.C.; Niebles, J.C.; Ghanem, B. Daps: Deep action proposals for action understanding. In Proceedings of the European Conference on Computer Vision. Amsterdam, The Netherlands, 11–14 October 2016; pp. 768–784.
39. Richard, A.; Gall, J. Temporal action detection using a statistical language model. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 3131–3140.
40. Caba Heilbron, F.; Carlos Niebles, J.; Ghanem, B. Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 1914–1923.
41. Mikolov, T.; Karafiát, M.; Burget, L.; Černockỳ, J.; Khudanpur, S. Recurrent neural network based language model. In Proceedings of the Eleventh Annual Conference of the International Speech Communication Association, Chiba, Japan, 26–30 September 2010.
42. Lin, T.; Zhao, X.; Su, H.; Wang, C.; Yang, M. BSN: Boundary Sensitive Network for Temporal Action Proposal Generation. *arXiv* **2018**, arXiv:1806.02964.
43. Zoph, B.; Vasudevan, V.; Shlens, J.; Le, Q.V. Learning transferable architectures for scalable image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 8697–8710.
44. Chen, Y.; Kalantidis, Y.; Li, J.; Yan, S.; Feng, J. Multi-fiber networks for video recognition. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 352–367.
45. Soomro, K.; Zamir, A.R.; Shah, M. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv* **2012**, arXiv:1212.0402.
46. Zach, C.; Pock, T.; Bischof, H. A duality based approach for realtime TV-L 1 optical flow. In *Proceedings of the Joint Pattern Recognition Symposium*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 214–223.
47. Graves, A. Supervised sequence labelling. In *Supervised Sequence Labelling with Recurrent Neural Networks*; Springer: Cham, Switzerland, 2012; pp. 5–13.
48. Gao, J.; Yang, Z.; Nevatia, R. Cascaded boundary regression for temporal action detection. *arXiv* **2017**, arXiv:1705.01180.