

Online temporal action detection method and device

Technical field

The present application relates to the field of computer vision, and in particular, to an online temporal action detection method and an online temporal action detection apparatus.

5

Background

As one of the sub-fields of computer vision (Computer Vision), video analysis technologies have developed rapidly in recent years. Prior to this, for a video analysis problem, a manual setting rule is usually used to find a key point in a screen, and information
10 such as a gradient and an optical flow of a change in position of the key point is analyzed by a machine learning (Machine Learning) method. Unlike two-dimensional images, a video consists of a series of ordered frames, and thus additionally has a time dimension. If a sequence relationship of the frames in the video is ignored, and the video is regarded as a collection of images only, there may be a problem that opening a door and closing a door, and
15 picking up and putting down cannot be distinguished and that can be solved by necessarily considering the frame association order. Therefore, one of the highlights of video analysis is to learn connections between frames.

Temporal action proposal generation in the field of video analysis may locate temporal regions that may contain specific human actions or events in an untrimmed video, and is
20 applicable to scenes such as video segmentation and video key segment selection. In a specific implementation, the temporal action detection needs to synchronously analyze the spatial content of the video on the image and to refine the video to a frame-level temporal relationship, and involves a plurality of steps such as video decoding, feature extraction, candidate action proposal generation, and proposal selection. Therefore, the temporal action
25 detection becomes challengeable and requires integration of various methods and technologies.

Summary

Embodiments of the present application provide an online temporal action detection method, an online temporal action detection apparatus, an online temporal action detection device, and a computer readable medium.

5 According to a first aspect, an online temporal action detection method is provided, and the method includes:

obtaining a video online and extracting an initial feature vector for action detection from the video; performing candidate action proposal generation processing, constructing a candidate action proposal according to the initial feature vector, and generating a confidence score associated with the candidate action proposal, wherein the confidence score represents a
10 degree of matching between the candidate action proposal and the ground truth action proposal in the video; and performing action proposal filtering processing, to filter out a redundant action proposal from the candidate action proposal, and output a generated action proposal;

where in the performing the generation processing on a candidate action proposal,
15 performing an online one-dimensional convolution operation for at least once based on the initial feature vector, and outputting a basic feature vector one time point by one time point; and respectively performing temporal evaluation processing and proposal evaluation processing based on the basic feature vector, wherein in the temporal evaluation processing, receiving a single basic feature vector one time point by one time point, and performing an
20 operation comprising a recurrent neural network, to generate a starting point probability sequence and an ending point probability sequence of the candidate action proposal one time point by one time point; in the proposal evaluation processing, cumulatively receiving a predetermined number of basic feature vectors generated one time point by one time point, performing an operation comprising sampling and convolution, and outputting a classification
25 confidence vector and a regression confidence vector; and calculating the confidence score based on the starting point probability sequence, the ending point probability sequence, the classification confidence vector, and the regression confidence vector.

In the online temporal action detection method of the present application, operations such as a lightweight online one-dimensional convolution, a long short-term memory network,
30 and a boundary matching layer are used, so that a probability that a current time point is used as a proposal boundary can be inferred along with real-time incoming video streams, and possibility that each time period using the current time point as an ending point is used as a proposal interval can be investigated. Compared with the prior art, the online temporal action detection method of the present application is more accurate in estimating a confidence score

of a candidate action proposal. In an actual application scenario of online temporal action detection, in order to control redundancy, when too many action proposals cannot be generated at each time point t , a higher recall rate can be achieved with fewer proposals. On the other hand, an online boundary matching network has less than one tenth of the number of parameters compared with an (offline) boundary matching network (BMN) with a similar structure. In a case of a high requirement for real-time performance, a millisecond-level single-step calculation time can also be achieved without relying on an accelerated operation of a video card, and is four to five times faster than that of a baseline that is an online boundary sensitive network. Even in an end-to-end environment in which a computing capability of a platform is limited and it is necessary to extract features from the generated video stream in real time and then perform action detection, the online action detection method may use an application scenario requires a finer granularity in a time dimension.

In a possible implementation of the first aspect, the online one-dimensional convolution operation includes: inputting the initial feature vector into a first buffer size one time point by one time point, where the first feature butter can store k initial feature vectors arranged one time point by one time point, and k is a size of a convolution kernel of the online one-dimensional convolution operation

In a possible implementation of the first aspect, wherein inputting the initial feature vector to a first feature buffer one time point by one time point further includes: inputting the initial feature vector to a first feature buffer one time point by one time point, where the first feature buffer can store k initial feature vectors arranged one time point by one time point, k is a size of a convolution kernel of the online one-dimensional convolution operation; and the online one-dimensional convolution operation further includes: performing a convolution operation on the k initial feature vectors in the first feature buffer by using the convolutional kernel, and obtaining the basic feature vector as an output result.

In a possible implementation of the first aspect, the temporal evaluation process includes: receiving a single basic feature vector one time point by one time point and performing an operation including a recurrent neural network, where processing on the basic feature vector at a current time point by the recurrent neural network combines a result of processing on the basic feature vector at a previous time point.

In a first possible implementation of the first aspect, the recurrent neural network further includes:

the recurrent neural network being a long short-term memory network; receiving, by the long short-term memory network, a single basic feature vector, an original hidden state h_{t-1}

and an original cell state c_{t-1} obtained from a long short-term memory network of a previous time point, and calculating and outputting a current new hidden state h_t based on the basic feature vector, the original hidden state h_{t-1} and the original cell state c_{t-1} ; and the temporal evaluation process further includes: inputting the new hidden state h_t into a fully connected layer one time point by one time point, and outputting the starting point probability sequence and the ending point probability sequence by using an activation function, where the starting point probability consists of a probability $p_{s,t}$ that a time point t is used as a starting point of the candidate action proposal, and the ending point probability consists of a probability $p_{e,t}$ that the time point t is used as an ending point of the candidate action proposal.

In a possible implementation of the first aspect, the method further includes setting an initial hidden state h_0 and an initial cell state c_0 to be 0.

In a possible implementation of the first aspect, in the proposal evaluation process, inputting the basic feature vector to a second feature buffer one time point by one time point, where the second feature buffer cumulatively receives and saves a predetermined number of basic feature vectors one time point by one time point.

In a possible implementation of the first aspect, inputting the basic feature vector to a second feature buffer one time point by one time point further includes: accumulatively receiving, by the second feature buffer, a predetermined number of basic feature vectors generated one time point by one time point, and storing the basic feature vectors as a basic feature sequence $S_F \in \mathbb{R}^{C \times D}$; the proposal evaluation process further includes: inputting the basic feature sequence and a sampling weight matrix $W \in \mathbb{R}^{N \times D}$ to a boundary matching layer to perform point multiplication, and obtaining and outputting a boundary matching matrix $M \in \mathbb{R}^{C \times N}$ having a sampling duration N and a time region length from 1 to D , and using the time point t as an ending point; and inputting the boundary matching matrix into a convolutional neural network including at least one layer to perform a convolution operation, and outputting the classification confidence vector $M_{cc} \in \mathbb{R}^D$ and the regression confidence vector $M_{cr} \in \mathbb{R}^D$;

where C denotes the number of channels of the basic feature vector, D denotes the length of the longest candidate action proposal in the basic feature sequence, the sampling weight matrix denotes a weight of the basic feature sequence S_F in the sampling duration N , M_{cc} denotes the probability of whether the candidate action proposal having a length from 1 to D and using a current time as an ending point is an action proposal, and M_{cr} denotes the probability that the candidate action proposal having a length from 1 to D and using a current

time as an ending point is close to an ground truth action proposal.

In a possible implementation of the first aspect, the at least one layer of convolutional neural network may include four layers of convolutional layers, where the first layer is a two-dimensional convolutional layer, and acts on two dimensions of the sampling duration N and the length D of the boundary matching matrix to integrate information of sampling points; both the second layer and the fourth layer are one-dimensional convolution layers, and act on a dimension of the length D to integrate information of channels; and the third layer is a one-dimensional convolutional layer, and acts on a dimension of the length D to integrate information of lengths of adjacent proposals.

In a possible implementation of the first aspect, the constructing the candidate action proposal further includes: determining that the time point t is a candidate action starting point, and adding the time point t of the candidate action starting point to a list of candidate action starting points; determining that the time point t is a candidate action ending point, selecting, from the list of candidate action starting points, the candidate starting point that can form the candidate action proposal having a length smaller than a maximum length D with the time point t, and outputting a set of candidate action proposals $\Phi_c = \{\phi_i = [t_{s,i}, t]\}_{i=1}^{N_c}$; where ϕ_i denotes an i^{th} candidate action proposal, N_c denotes the number of candidate action proposals, and $t_{s,i}$ denotes a start time of the i^{th} candidate action proposal.

In a possible implementation of the first aspect, the confidence score is calculated by the following formula:

$$\text{score}_i = p_{s,t_{s,i}} \times p_{e,t} \times \sqrt{M_{cc,t}[t - t_{s,i} + 1] \times M_{cr,t}[t - t_{s,i} + 1]},$$

where $t - t_{s,i} + 1$ denotes a length of the candidate action proposal ϕ_i , $p_{s,t}$ and $p_{e,t}$ respectively denote probabilities that a time point t is used as a starting point and an ending point of the candidate action proposal, $M_{cc,t}$ denotes the probability of whether the candidate action proposal having a length from 1 to D and using a time point t as an ending point is an action proposal, and $M_{cr,t}$ denotes the probability that the candidate action proposal having a length from 1 to D and using a time point t as an ending point is close to the ground truth action proposal.

In a possible implementation of the first aspect, the initial feature vector includes temporal structure information.

According to a second aspect, an online temporal action detection apparatus is provided, including: a video obtaining unit, configured to obtain a video online and extracting an initial feature vector for action detection from the video; a candidate action proposal generation unit,

configured to construct a candidate action proposal based on the initial feature vector, and generate a confidence score associated with the candidate action proposal based on the candidate action proposal and a corresponding ground truth action proposal in the video, where the confidence score represents a degree of matching between the candidate action
5 proposal and the ground truth action proposal in the video; and an action proposal filtering unit, configured to filter out a redundant action proposal from the candidate action proposal and output a generated action proposal;

where in the candidate action proposal generating unit, an online one-dimensional convolution operation subunit is configured to perform an online one-dimensional
10 convolution operation for at least one time based on the initial feature vector, and output a basic feature vector one time point by one time point; and respectively input the basic feature vector into a temporal evaluation subunit and a proposal evaluation subunit to perform an operation, where the temporal evaluation subunit is configured to receive a single basic feature vector one time point by one time point, and perform an operation comprising a
15 recurrent neural network, to generate a starting point probability sequence and an ending point probability sequence of the candidate action proposal one time point by one time point; the proposal evaluation subunit is configured to cumulatively receive a predetermined number of basic feature vectors generated one time point by one time point, perform an operation comprising sampling and convolution, and output a classification confidence vector and a
20 regression confidence vector; and a confidence score calculation subunit is configured to calculate the confidence score based on the starting point probability sequence, the ending point probability sequence, the classification confidence vector, and the regression confidence vector.

In a possible implementation of the second aspect, in the online one-dimensional
25 convolution operation subunit, the initial feature vector is inputted to a first feature buffer one time point by one time point, where the first feature buffer can store k initial feature vectors arranged one time point by one time point, and k is a size of a convolution kernel of the online one-dimensional convolution operation.

In a possible implementation of the second aspect, the online one-dimensional
30 convolution operation subunit further includes: the first feature buffer, used to receive the initial feature vector one time point by one time point and delete the initial feature vector first inputted into the first feature buffer, to ensure that k initial feature vectors most recently inputted one time point by one time point are stored in the first feature buffer, where k represents the size of the convolution kernel; and a one-dimensional convolution layer, used

to perform a convolution operation on the k initial feature vectors in the first feature buffer, and obtain the basic feature vector as an output result.

In a possible implementation of the second aspect, the temporal evaluation subunit includes: receiving a single basic feature vector one time point by one time point and performing an operation including a recurrent neural network, where processing on the basic feature vector at a current time point by using the recurrent neural network combines a result of processing on the basic feature vector at a previous time point.

In a possible implementation of the second aspect, the temporal evaluation subunit further includes: a long short-term memory network, the recurrent neural network being a long short-term memory network, used to receive a single basic feature vector, an original hidden state h_{t-1} and an original cell state c_{t-1} obtained from a previous long short-term memory network, and calculating and outputting a current new hidden state h_t based on the basic feature vector, the original hidden state h_{t-1} and the original cell state c_{t-1} ; and a fully connected layer, used to receive the new hidden state h_t one time point by one time point, and output the starting point probability sequence and the ending point probability sequence by using an activation function, wherein the starting point probability consists of a probability $p_{s,t}$ that a time point t is used as a starting point of the candidate action proposal, and the ending point probability consists of a probability $p_{e,t}$ that the time point t is used as an ending point of the candidate action proposal.

In a possible implementation of the second aspect, the apparatus further includes setting an initial hidden state h_0 and an initial cell state c_0 to be 0.

In a possible implementation of the second aspect, in the proposal evaluation subunit, inputting the basic feature vector into a second feature buffer one time point by one time point, where the second feature buffer cumulatively receives and stores a predetermined number of basic feature vectors generated one time point by one time point.

In a possible implementation of the second aspect, the proposal evaluation subunit further includes: the second feature buffer, used to accumulatively receive a predetermined number of basic feature vectors generated one time point by one time point and store the basic feature vectors as a basic feature sequence $S_F \in \mathbb{R}^{C \times D}$; a boundary matching layer, used to perform point multiplication on the basic feature sequence and a sampling weight matrix $W \in \mathbb{R}^{N \times D}$, and obtain and output a boundary matching matrix $M \in \mathbb{R}^{C \times N}$ having a sampling duration N and a time region length from 1 to D , and using the time point t as an ending point; and a convolutional neural network, the convolutional neural network including at least one layer, used to input the boundary matching matrix to perform a convolution operation, and

output the classification confidence vector $M_{cc} \in \mathbb{R}^D$ and the regression confidence vector $M_{cr} \in \mathbb{R}^D$;

where C denotes the number of channels of the basic feature vector, D denotes the length of the longest candidate action proposal in the basic feature sequence, the sampling weight matrix denotes a weight of the basic feature sequence S_F in the sampling duration N , M_{cc} denotes the probability of whether the candidate action proposal having a length from 1 to D and using a current time as an ending point is an action proposal, and M_{cr} denotes the probability that the candidate action proposal having a length from 1 to D and using a current time as an ending point is close to an ground truth action proposal.

In a possible implementation of the second aspect, the at least one layer of convolutional neural network may include four layers of convolutional layers, where the first layer is a two-dimensional convolutional layer, and acts on two dimensions of the sampling duration N and the length D of the boundary matching matrix to integrate information of sampling points; both the second layer and the fourth layer are one-dimensional convolution layers, and act on a dimension of the length D to integrate information of channels; and the third layer is a one-dimensional convolutional layer, and acts on a dimension of the length D to integrate information of lengths of adjacent proposals.

In a possible implementation of the second aspect, the constructing a candidate action proposal further includes: determining that the time point t is a candidate action starting point, and adding the time point t of the candidate action starting point to a list of candidate action starting points; determining that a time point t is a candidate action ending point, selecting, from the list of candidate action starting points, the candidate starting point that can form the candidate action proposal having a length smaller than a maximum length D with the time point t , and outputting a set of candidate action proposals $\Phi_c = \{\phi_i = [t_{s,i}, t]\}_{i=1}^{N_c}$; where ϕ_i denotes an i^{th} candidate action proposal, N_c denotes the number of candidate action proposals, and $t_{s,i}$ denotes a start time of the i^{th} candidate action proposal.

In a possible implementation of the first aspect, the confidence score is calculated by the following formula:

$$\text{score}_i = p_{s,t_{s,i}} \times p_{e,t} \times \sqrt{M_{cc,t}[t - t_{s,i} + 1] \times M_{cr,t}[t - t_{s,i} + 1]},$$

where $t - t_{s,i} + 1$ denotes a length of the candidate action proposal ϕ_i , $p_{s,t}$ and $p_{e,t}$ respectively denote probabilities that a time point t is used as a starting point and an ending point of the candidate action proposal, $M_{cc,t}$ denotes the probability of whether the candidate action proposal having a length from 1 to D and using a time point t as an ending point is an

action proposal, and $M_{cr,t}$ denotes the probability that the candidate action proposal having a length from 1 to D and using a time point t as an ending point is close to the ground truth action proposal.

In a possible implementation of the second aspect, the initial feature vector includes temporal structure information.

In the online temporal action detection apparatus of the present application, operations such as a lightweight online one-dimensional convolution, a long short-term memory network, and a boundary matching layer are used, so that a probability that a current time point is used as a proposal boundary can be inferred along with real-time incoming video streams, and possibility that each time period using the current time point as an ending point is used as a proposal interval can be investigated. Compared with the prior art, the online temporal action detection apparatus of the present application is more accurate in estimating a confidence score of a candidate action proposal. In an actual application scenario of online temporal action detection, in order to control redundancy, when too many action proposals cannot be generated at each time point t, a higher recall rate can be achieved with fewer proposals. On the other hand, an online boundary matching network has less than one tenth of the number of parameters compared with an (offline) boundary matching network (BMN) with a similar structure. In a case of a high requirement for real-time performance, a millisecond-level single-step calculation time can also be achieved without relying on an accelerated operation of a video card, and is four to five times faster than that of a baseline that is an online boundary sensitive network. Even in an end-to-end environment in which a computing capability of a platform is limited and it is necessary to extract features from the generated video stream in real time and then perform action detection, the online action detection apparatus may use an application scenario requires a finer granularity in a time dimension.

According to a third aspect, an online temporal action detection device is provided, including:

- a memory, configured to store an instruction executed by one or more processors of a system, and

- a processor, being one of the processors of the system, configured to execute the instruction to implement the temporal action detection method according to any one of the possible implementations of the first aspect.

According to a fourth aspect, a computer readable storage medium encoded with a computer program is provided, where an instruction is stored on the computer readable medium, and when being executed on a computer, the instruction enables the computer to

perform the temporal action detection method according to any one of the possible implementations of the first aspect.

Brief description of drawings

FIG. 1 is a block diagram of an online temporal action detection system according to
5 some embodiments of the present application;

FIG. 2 is an application scenario for online temporal action detection according to some
embodiments of the present application;

FIG. 3 is a structural block diagram of an electronic device according to some
embodiments of the present application;

10 FIG. 4 is a flowchart of an online temporal action detection method according to some
embodiments of the present application;

FIG. 5 is a schematic diagram of online extraction of initial feature vectors from a video,
according to some embodiments of the present application;

15 FIG. 6 is a flowchart of a candidate action proposal generation step according to some
embodiments of the present application;

FIG. 7 is a structural block diagram of an online one-dimensional convolution operation
according to some embodiments of the present application;

FIG. 8 is a diagram of probabilities of a starting point and an ending point outputted by
temporal evaluation processing according to some embodiments of the present application;

20 FIG. 9 is a block diagram of a boundary matching layer according to some embodiments
of the present application;

FIG. 10 is a diagram of a confidence vector outputted by proposal evaluation processing
according to some embodiments of the present application; and

25 FIG. 11 is a schematic diagram of an online temporal action detection apparatus
according to some embodiments of the present application.

Detailed description of embodiments

Illustrative embodiments of the present application include, but are not limited to, an
online temporal action detection method, an online temporal action detection apparatus, an

online temporal action detection apparatus, and a computer readable medium.

FIG. 1 is a block diagram of an online temporal action detection system according to some embodiments of the present application. As shown in FIG. 1, the system includes a front end device 220 and an electronic device 240. The front end device 220 and the electronic device 240 are communicatively connected, for example, through a wired network, WIFI, ZigBee, or Bluetooth.

The front end device 220 may be a surveillance camera, a smart home device, a chat robot, a desktop computer, a smart phone, or the like. The front end device 220 is capable of capturing a video, generating a video, storing a video, or downloading a video. The front end device 220 provides a video to be processed to the electronic device 240 in real time.

The electronic device 240 is provided with an online temporal action detection model. The model is used to perform online temporal evaluation and proposal evaluation on the video, and output a generated action proposal of the video according to an evaluation result. The generated action proposal may be used for subsequent action detection, video analysis, security alarms, and the like.

Temporal action detection is also known as temporal action proposal generation. For a video, several candidate temporal proposals that may contain specific human actions or events (collectively referred to as "actions") are found in the video. Temporal action detection sometimes also stands for Temporal Action Detection (or Temporal Action Localization, having the same meaning). The main difference between the two is that Temporal Action Proposal Generation only selects action proposals, but Temporal Action Detection also needs to give category information of action proposals. For the Temporal Action Detection method that includes two steps: first selecting action proposals and then performing classification, Temporal Action Proposal Generation can be regarded as the first step. In the present application, temporal motion detection specifically refers to Temporal Action Proposal Generation.

Temporal action detection has many application scenarios. In real life, a video is often untrimmed. A detailed description is given below with reference to a specific scenario of a video of playing a musical instrument shown in FIG. 2. As shown in FIG. 2, a time interval of the entire video is from 0.0s to 119.2s, and contents of the video include contents such as a description of a playing track, performance of a player, and a list of producers. A key part is a ground truth action proposal from a ready-to-play gesture of the player to the end of playing and putting down a musical instrument, and a start and end time is from 13.8s to 97.9s. In order to make the proposal receive more attention on a video website and facilitate labeling of

a video category or other analysis work, a playing video is obtained online and an initial feature vector is extracted for action detection from the video in the present application. A candidate action proposal is constructed according to the initial feature vector, and a confidence score associated with the candidate action proposal is generated according to the candidate action proposal and a corresponding ground truth action proposal from a ready-to-play gesture to the end of playing and putting down a musical instrument in the video. Action proposal filtering processing is performed to filter out a redundant action proposal from the candidate action proposal and output a generated action proposal. According to the temporal action detection method in the present application, a proposal in which a player plays is labeled, and a generated action proposal having a start and end time is from 14.2s to 98.0s is relatively close to a ground truth action proposal, so that an action proposal that may contain a specific human action or an event can be located online in real time in the untrimmed video. In addition, when analysis tasks such as human body action recognition, action analysis of a teacher teaching a class in an educational scene, performance analysis of a student taking a class in an educational scene, extraction of an essence video from a long video, and key action detection in a short video are performed to substitute for a wearable device, proposals of actions can be found from the video by using the temporal action detection method, thereby facilitating subsequent processing on the actions one by one.

Heretofore, most of the temporal action detection methods are offline, that is, temporal action detection is performed after a complete video is obtained. An offline temporal action detection method needs to comprehensively consider information of the entire video. For example, in a video of playing a musical instrument, even if an action of putting down the musical instrument by a player is not concerned, it can be inferred that a performance has ended at a previous time by using a list of producers appearing at a later time. However, in some cases in which there is a high requirement for real-time performance, the offline temporal action detection method in which a complete video needs to be obtained and then processing is performed becomes inapplicable. For example, in processing of a surveillance video, it is necessary to detect a suspicious person action on a screen in real time online, and to give an action proposal immediately at the end of the action (within a very short millisecond-level delay time for the operation of the method), so as to remind a security personnel to pay attention in time, or to perform downstream tasks such as action recognition.

Therefore, the online temporal action detection method needs to provide the action proposal in real time as the video proceeds. The most significant difference from the offline method is that a video content after a current time cannot be obtained, and only an action

proposal using the current time as an ending point can be provided by analyzing a previous video content. A Temporal Recurrent Network is a rare example in which an action proposal can be detected online from a video. In an implementation process, the support of action category labels is required, and therefore the method cannot process a video containing an
5 unseen action category, and a method for generating a proposal based on a prediction result of combined action categories cannot distinguish a plurality of actions occurring simultaneously and having time overlaps in the video. For this reason, there is a need to get rid of the limitation of the action category and focus more on spatial and temporal variations of the video in order to make the method applicable to a wide variety of video contents. To this end,
10 the present application provides an online temporal action detection method having high accuracy and real-time performance, and being capable of taking both effect and efficiency into account to meet such needs.

It can be understood that the above description of an application scenario for performing online temporal action detection is merely exemplary and not limiting.

15 FIG. 3 is a structural block diagram of an electronic device 100 according to some embodiments of the present application. Specifically, as shown in FIG. 3, the electronic device 100 includes one or more processors 104, a system control logic 108 connected to at least one of the processors 104, a system memory 112 connected to the system control logic 108, a non-volatile memory (NVM) 116 connected to the system control logic 108, and a
20 network interface 120 connected to the system control logic 108.

In some embodiments, the processor 104 may include one or more single-core or multi-core processors. In some embodiments, the processor 104 may include any combination of a general-purpose processor and a dedicated processor (for example, a graphics processor, an application processor, and a baseband processor). In an embodiment in which the
25 electronic device 100 uses an evolved node B (evolved Node B, eNB) or a radio access network (radio access network, RAN) controller, the processor 104 may be configured to perform various corresponding embodiments.

In some embodiments, the processor 104 may be configured to perform online temporal action detection. Specifically, for example, in a process of processing an image by each layer
30 of a neural network model, the processor 104 continuously adjusts the neural network model according to the obtained image information until a candidate action proposal with higher accuracy is obtained. The processor 104 may also periodically update the neural network model, so as to better adapt to changes in various actual demands on the neural network model.

In some embodiments, the system control logic 108 may include any suitable interface controller to provide any suitable interface for at least one of the processors 104 and/or any suitable device or component in communication with the system control logic 108.

5 In some embodiments, the system control logic 108 may include one or more memory controllers to provide an interface connected to the system memory 112. The system memory 112 may be used to load and store data and/or an instruction. In some embodiments, the memory 112 of the electronic device 100 may include any suitable volatile memory, such as a suitable dynamic random access memory (DRAM). In some embodiments, the system memory 112 may be used to load or store an instruction for implementing the online temporal
10 action detection, or the system memory 112 may be used to load or store an instruction for implementing an application program for the online temporal action detection based on a neural network. The processor 104 can implement the online temporal action detection method according to the embodiments of the present application by running the instruction.

The NVM/memory 116 may include one or more tangible, non-transitory
15 computer-readable media for storing data and/or instructions. In some embodiments, the NVM/memory 116 may include any suitable non-volatile memory such as a flash memory and/or any suitable non-volatile storage device, such as at least one of a hard disk drive (HDD), a compact disk (CD) drive, and a digital versatile disc (DVD) drive. The NVM/memory 116 may further be used to store the neural network model used in the above
20 online temporal action detection process.

The NVM/memory 116 may include a part of storage resources on an apparatus on which the electronic device 100 is installed, or may be accessed by the device, but not necessarily a part of the device. For example, the NVM/memory 116 can be accessed by using a network via the network interface 120.

25 In particular, the system memory 112 and the NVM/memory 116 may include: a temporary copy and a permanent copy of an instruction 124, respectively. The instruction 124 may include an instruction that causes the electronic device 100 to implement the online temporal action detection method in the present application when executed by at least one of the processors 104. In some embodiments, the instruction 124, hardware, firmware, and/or
30 software components thereof may additionally/alternatively be placed in system control logic 108, the network interface 120, and/or the processor 104.

The network interface 120 may include a transceiver configured to provide a radio interface for the electronic device 100 to further communicate with any other suitable devices (such as, a front-end module and an antenna) by using one or more networks. In some

embodiments, the network interface 120 may be integrated into other components of the electronic device 100. For example, the network interface 120 may be integrated into at least one of the processor 104, the system memory 112, the NVM/memory 116, and a firmware device (not shown) with an instruction. When at least one of the processors 104 executes the instruction, the electronic device 100 implements the online temporal action detection method in the present application.

The network interface 120 may further include any suitable hardware and/or firmware to provide a multiple-input multiple-output radio interface. For example, the network interface 120 may be a network adapter, a wireless network adapter, a telephone modem, and/or a wireless modem.

In some embodiments, at least one of the processors 104 may be packaged with logics of one or more controllers for the system control logic 108 to form a system in package (SiP). In some embodiments, at least one of the processors 104 may be integrated on the same die with the logics of one or more controllers for the system control logic 108 to form a system on chip (SoC).

The electronic device 100 may further include: an input/output (I/O) device 132. The I/O device 132 may include a user interface, so that the user can interact with the electronic device 100. A peripheral component can also interact with the electronic device 100 by using the design of a peripheral component interface. In some embodiments, the electronic device 100 further includes a sensor for determining at least one of environmental conditions and location information related to the electronic device 100.

According to some embodiments of the present application, an online temporal action detection method 400 is provided. As shown in FIG. 4, the method 400 includes the following steps:

Step 401: Obtain a video online and extract an initial feature vector for action detection from the video.

Specifically, in step 401, for example, in the processing of a real-time surveillance video, it is necessary to detect in real time a suspicious person action in a screen, and therefore it is necessary to obtain a real-time video online, that is, to obtain a video generated in real time one time point by one time point, and the video uses a current time point as an ending point. After the video is obtained, feature extraction is performed on the video. Specific operations are as follows: a last Fully Connected Layer (FC Layer) and Global Temporal Pooling are removed based on a frame of the action recognition method, and a convolutional and/or recurrent neural network are/is reserved, thereby extracting a video feature including a

temporal sequence structure, called an initial feature vector. The initial feature vector includes static features such as color features, texture features, shape features, sound features, and action features including temporal structures.

In the present application, $X = \{x_i\}_{i=0}^{l_v-1}$ is used to indicate a video segment, l_v indicates the number of frames of the video segment, x_i indicates a content of an i^{th} frame, and the content may include information such as an image RGB or an optical flow according to the input used in the feature extraction step 401. FIG. 5 is a schematic diagram of online extraction of an initial feature vector from a video according to some embodiments of the present application. After the feature extraction step shown in FIG. 5, a sequence of initial feature vectors indicated by $F = \{f_j\}_{j=0}^{l_f-1}$, l_f indicates a sequence length, $f_j \in R^C$ indicates an initial feature vector of a j^{th} position in the sequence, and C indicates the Channel number of the initial feature vector. As shown in FIG. 5, $C=6$, $l_f=8$, that is, a video of a player playing an instrument is uniformly mapped to eight initial feature vectors having a channel number of 6 by feature extraction. Generally, features may be extracted proportionally from the video (for example, each σ frame in the original video corresponds to one of the feature vectors). A different σ may be used for each video according to a duration of the video, and the length l_f of the feature sequence may also be fixed by a Linear Interpolation method. In conclusion, it can be understood that the feature extraction processing is to uniformly map the content of the l_v frame of the original video frame onto the l_f initial feature vectors, and F is an abstract representation of the original video after scaling of a time scale. Therefore, for the subsequent temporal action detection method, each initial feature vector corresponds to a time length in the original video, that is, a minimum time unit. For convenience, T is used to indicate l_f in the present application, and the ordinal number of the initial feature sequence is referred to as a time point t without confusion.

Step 402: Perform a candidate action proposal generation process, construct a candidate action proposal, and generate a confidence score.

Specifically, in step 402, based on the initial feature vector extracted from the video, a probability that a time point t is used as a starting point and a probability that a time point t is used as an ending point of the candidate action proposal are obtained by temporal evaluation processing. Based on this, the candidate action proposal is constructed, and then the confidence score associated with the candidate action proposal is generated based on a classification confidence vector and a regression confidence vector obtained by proposal

evaluation processing. The confidence score indicates a degree of matching between the candidate action proposal and a ground truth action proposal. A specific implementation is described below in detail.

$\Phi_c = \{\phi_i = [t_{s,i}, t_{e,i}]\}_{i=1}^{N_c}$ is used to indicate a set of candidate action proposals,

5 $\Phi_g = \{\phi_i = [t_{s,i}, t_{e,i}]\}_{i=1}^{N_g}$ is used to indicate a set of Ground Truth action proposals, N_c indicates the number of candidate action proposals in the video, and N_g indicates the number of ground truth action proposals in the video, and $t_{s,i}$ and $t_{e,i}$ respectively indicate a start time and an end time of an i^{th} action proposal ϕ_i . The set Φ_g of ground truth action proposals can be covered as accurately and comprehensively as possible by constructing the

10 set Φ_c of candidate action proposals and determining the confidence score in the temporal action detection.

Step 403: Perform action proposal filtering processing, to filter out a redundant action proposal according to the confidence score, and output a generated action proposal.

Specifically, in step 403, in the temporal action detection, boundary probabilities of two

15 time points with a very short difference are likely to be very close, and consequently repeated and redundant proposals occur in the set of candidate action proposals.

A Non-Maximum Suppression (NMS) method is very commonly used in tasks such as filtering out a redundant candidate box or performing the candidate action proposal. The remaining proposals are removed by using a method in which a candidate action proposal

20 having a largest confidence score in similar candidate action proposals is reserved and the confidence score is set to be 0. However, there is a disadvantage that, in two candidate action proposals that are sufficiently close but both true actions, the candidate action proposal having a small confidence score is directly rule out, and therefore this method is also referred to as a hard non-maximum suppression (Hard-NMS) method. In the technical solution of the present

25 application, a soft non-maximum suppression method (Soft-NMS) derived from the non-maximum suppression method can be used to reserve the possibility that the candidate action proposals close to each other are selected. The procedure is as follows:

1. Assume that a set of generated action proposals finally reserved is $\Phi_p = \{ \}$.

2. Select a candidate action proposal with a largest confidence score, denoted as

30 $\phi_p = [t_{s,p}, t_{e,p}, score_p]$ from a set of candidate action proposals $\Phi_c = \{\phi_i = [t_{s,i}, t_{e,i}, score_i]\}_{i=1}^{N_c}$. ϕ_p is selected from Φ_c into Φ_p .

3. Traverse all remaining candidate action proposals in Φ_c , and if the intersection over union of action proposals ϕ_i and ϕ_p meets that $\text{IoU}(\phi_i, \phi_p) > 0.8$, change a confidence score ϕ_i to:

$$\text{score}_i = \text{score}_i \times \exp\left(\frac{-\text{IoU}(\phi_i, \phi_p)^2}{0.75}\right), \dots (1)$$

5 4. Repeat steps 2-3, and when the number of reserved proposals in Φ_p reaches the preset N_p or all proposals in Φ_c have been selected into Φ_p , end filtering, and return Φ_p .

Finally, all the proposals in the set Φ_p are generated action proposals that use the time point t as an ending point given by the online temporal action detection method. When the start time and end time (in this case, that is, the ordinal number of the corresponding feature vector) of the generated action proposal are mapped to a real time, the midpoint of a time interval $r_i = [i \times \frac{t_v}{T}, (i+1) \times \frac{t_v}{T}]$ corresponding to a feature sequence i is used, that is, the real time $t_{real,i} = (i+0.5) \times \frac{t_v}{T}$, where t_v is the original video duration, and T is the length of the feature sequence.

15 A specific process of candidate action proposal generation performed in step 402 is further explained below with reference to FIG. 6. As shown in FIG. 6, the following steps may be illustratively included:

Step 4021: Perform an online one-dimensional convolution operation for at least one time based on the initial feature vector, and output a basic feature vector one time point by one time point.

Specifically, in step 4021, in a real-time video environment, it can be considered that the initial feature vector in a video feature sequence is generated time-one time point by one time point and inputted to an online one-dimensional convolutional layer. Therefore, a convolution operation cannot be performed on the complete sequence. To this end, the present application proposes a technical solution of online one-dimensional convolution (Online Conv1d). FIG. 7 is a block diagram of an online one-dimensional convolution operation according to some embodiments of the present application. As shown in FIG. 7, the online one-dimensional convolution includes a first feature buffer and a one-dimensional convolution, and the first feature buffer may store k initial feature vectors inputted most recently, where k is a size of a

convolution kernel. As time goes by, each time an initial feature vector extracted from a video corresponding to a new time point is generated and inputted into the online one-dimensional convolution, the initial feature vector is put into the tail of the first feature buffer, and an initial feature vector that is first put into the first feature buffer is deleted, so as to ensure that

5 k initial feature vectors that are stored in the first feature buffer are most recently inputted one time point by one time point. Then, a single-step and non-sliding convolution operation is performed on the initial feature vectors in the first feature buffer by using the convolution kernel, and a basic feature vector is finally generated as an output result. The online one-dimensional convolution operation can meet a requirement of real-time online calculation

10 with the extraction of the basic feature vector one time point by one time point and can achieve the same effect as the conventional one-dimensional convolution. Specifically, an online one-dimensional convolutional neural network optionally includes two layers. As shown in Table 1, a value of the size k of the convolution kernel is 3, that is, the first buffer stores the most recent three initial feature vectors, and a convolution operation is performed

15 on every three adjacent initial feature vectors as the initial feature vectors are inputted to the first feature buffer one time point by one time point, and a basic feature vector for every three adjacent initial feature vectors is generated, so that the basic feature vectors are outputted one time point by one time point.

The above value of k is merely illustrative description of the embodiments of the present application. The value of k is determined by an actual application. For example, a convolution

20 kernel size k that is 5 or 7 may be selected in other cases.

Table 1

Layer	Kernel Size	Output Channel	Activation	Output Size
Online Conv1d 1	3	256	ReLU	256
Online Conv1d 2	3	128	ReLU	128

25 A quantitative representation of an online one-dimensional convolutional layer operation is described below. Assuming that the number of input channels of an online one-dimensional convolution layer is C_{in} , the number of output channels is C_{out} , and the size of the convolution kernel is k, the layer has two sets of parameters: a weight $W \in \mathbb{R}^{C_{out} \times C_{in} \times k}$ and a bias $B \in \mathbb{R}^{C_{out}}$. Initialization of the parameters of the online one-dimensional convolutional layer follows the

30 method of initialization of parameters of a conventional convolutional convolution, and values of elements of a weight matrix and a bias vector are randomly selected from

$U(-\sqrt{\frac{1}{C_{in} \times k}}, \sqrt{\frac{1}{C_{in} \times k}})$, where $U(a, b)$ is a uniform distribution from a to b . For input features (including features in the buffer) $F_{in} \in \mathbb{R}^{C_{in} \times k}$ of the online one-dimensional convolution on this layer, output features are as follows:

$$F_{out}[c_{out}] = B[c_{out}] + \sum_{c_{in}=0}^{C_{in}-1} \sum_{j=0}^{k-1} W[c_{out}, c_{in}, j] \times F_{in}[c_{in}, j] (c_{out} = 0, 1, \dots, C_{out}-1) \dots (2)$$

5 The calculated F_{out} undergoes a rectified linear unit function (ReLU) $\text{ReLU}(x) = \max(0, x)$, and an application range $[0, +\infty)$ of depth learning in video analysis is outputted. Calculation in the above operation is simple, and can be completed by only one-time matrix multiplication and one-time matrix addition, so that a requirement of online temporal action detection for real-time performance can be met. The online one-dimensional convolution operation further
10 processes the initial feature vector generated at the feature extraction stage, and enlarges a receptive field of the feature at the same time, so that the generated basic feature vector can contain more information about temporal sequence information associated with the adjacent video frames.

15 Step 4022: In the temporal evaluation processing, a single basic feature vector is received one time point by one time point, and an operation including a recurrent neural network is performed, to generate a starting point probability sequence and an ending point probability sequence of a candidate action proposal one time point by one time point.

Specifically, in step 4022, unlike most of the frame/video segment-based temporal action detection methods based on in which use a convolutional neural network is used to obtain the
20 starting point probability sequence and the ending point probability sequence, the temporal evaluation processing in the present application includes a long short-term memory network and a fully connected layer, as shown in Table 2. Compared with an offline temporal action detection method, the online temporal action detection cannot learn the information after a current time. Therefore, in order to utilize history information of a longer time, the recurrent
25 neural network is used in the present application to replace the convolutional neural network. Each basic feature vector outputted by the online one-dimensional convolution operation is inputted to a long short-term memory cell (LSTM Cell). The long short-term memory cell simultaneously receives a hidden state and a cell state obtained last time, calculates a current new hidden state and a new cell state, and outputs the new hidden state as a feature. An initial
30 hidden state and an initial cell state are set to be 0. The hidden state enters the fully connected layer, and a probability $p_{s,t}$ that a time point is used as a starting point and a probability $p_{e,t}$

that a time point t is used as an ending point of the candidate action proposal are finally outputted.

5

Table 2

Layer	Hidden Size	Output Channel	Activation	Output Size
LSTM Cell	128	128		128
FC Layer		2	Sigmoid	2

Quantitative representations of operations of the long short-term memory network are as follows. Assuming that a feature vector inputted at a time point t is $F_t \in \mathbb{R}^{\text{Cin}}$, an original hidden state calculated last time is $h_{t-1} \in \mathbb{R}^S$, and an original cell state is $c_{t-1} \in \mathbb{R}^S$, where S is the length of the hidden state and the cell state. The operations of the long short-term memory network may be expressed as:

$$i_t = \sigma(W_{ii} \cdot F_t + b_{ii} + W_{hi} \cdot h_{t-1} + b_{hi}) \quad (3)$$

$$f_t = \sigma(W_{if} \cdot F_t + b_{if} + W_{hf} \cdot h_{t-1} + b_{hf}) \quad (4)$$

$$g_t = \tanh(W_{ig} \cdot F_t + b_{ig} + W_{hg} \cdot h_{t-1} + b_{hg}) \quad (5)$$

$$o_t = \sigma(W_{io} \cdot F_t + b_{io} + W_{ho} \cdot h_{t-1} + b_{ho}) \quad (6)$$

$$c_t = f_t * c_{t-1} + i_t * g_t \quad (7)$$

$$h_t = o_t * \tanh(c_t) \quad (8)$$

Where $A*B$ denotes the Hadamard product, that is, for vectors $A=\{a_i\}$ and $B=\{b_i\}$ of a same size, a vector $C=\{c_i\}$ of the same size is returned, so that a value of each element in C is the product of values of position elements corresponding to A and B , that is, $c_i = a_i \times b_i$.

$\sigma(x) = \frac{1}{1+e^{-x}}$ denotes a Sigmoid function, and an output range is $(0, 1)$; $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

denotes a hyperbolic tangent function, and an output range is $(-1, 1)$; i denotes an input gate for processing input information; g denotes a Cell Gate for controlling which input information is added to the new cell state c_t ; f denotes a Forget Gate for controlling which contents of the original cell state are retained and discarded; o denotes an output gate that processes and prepares output information. Each W and each b are a weight and a bias obtained when each gate respectively acts on the input basic feature vector F_t and the original

hidden state h_{t-1} , and the initialization method is that values of elements are randomly selected from $U(-\sqrt{\frac{1}{S}}, \sqrt{\frac{1}{S}})$. The new hidden state h_t that is ultimately used as the output feature is determined by the vector generated by the output gate and the new cell state together.

Then, the fully connected layer is used to integrate the new hidden state outputted by the long short-term memory network, and two values in the range (0, 1) are outputted by using the Sigmoid activation function, and respectively indicate the probabilities that the time point t is used the starting point and the time point t is used as the ending point of the candidate action proposal. The probability that the time point t is used as the starting point of the candidate action proposal can be understood as the probability that the time point or time interval of the video time corresponding to the t^{th} feature vector is the starting point. Generally, the function of the temporal evaluation operation is to cyclically receive, one time point by one time point, the basic feature vector outputted by the online one-dimensional convolution operation, and to generate, one time point by one time point, a starting point probability sequence and an ending point probability sequence of the candidate action proposal. For a video initial feature sequence of a length T , the output of the temporal evaluation processing is a starting point probability sequence $P_s = \{p_{s,i}\}_{i=0}^{T-1}$ and an ending point probability sequence $P_e = \{p_{e,i}\}_{i=0}^{T-1}$. FIG. 8 is a diagram of a starting point probability and an ending point probability outputted by temporal evaluation processing according to some embodiments of the present application. As shown in FIG. 8, a coordinate of a horizontal axis is a time value from 0 to a time point t , a coordinate of a vertical axis is a probability value from 0 to 1, a solid line indicates a probability $p_{s,t}$ that a current time is used as a starting point of a candidate action proposal, and a dashed line indicates a probability $p_{e,t}$ that a current time is used as an ending point of a candidate action proposal.

Step 4023: Perform proposal evaluation processing, cumulatively receive a predetermined number of basic feature vectors generated one time point by one time point, perform an operation including sampling and convolution, and output a classification confidence vector and a regression confidence vector.

Specifically, in step 4023, a second feature buffer, a boundary matching layer and a four-layer convolutional neural network are included, as shown in Table 3.

Table 3

Layer	Kernel Size	Output Channel	Activation	Output Size
BM Layer		128		$128 \times N \times D$
Conv2d	32, 1	512	ReLU	$512 \times 1 \times D$
Conv1d 1	1	128	ReLU	$128 \times D$
Conv1d 2	3	128	ReLU	$128 \times D$
Conv1d 3	1	2	Sigmoid	$2 \times D$

The second feature buffer accumulatively receives a predetermined number of basic feature vectors generated one time point by one time point. For example, the most recent D basic feature vectors are received and saved as a basic feature sequence $S_F \in \mathbb{R}^{C \times D}$. FIG. 9 is a structural block diagram of a boundary matching layer according to some embodiments of the present application. As shown in FIG. 9, the function of the boundary matching layer is to input the most recent D basic feature vectors (D denotes a length of a longest action proposal in the feature sequence, and a specific value depends on a length of a ground truth action proposal in a dataset) outputted by the online one-dimensional convolution operation and stored in the second feature buffer. These basic feature vectors are multiplied by a sampling weight matrix, and features within or near all intervals using the time point t as an ending point and having a length from 1 to D are sampled as a new feature matrix of a fixed length in a time dimension, so as to facilitate subsequent calculation.

For a proposal having a length d (that is, starting points $s=t-d+1$, $d=1, 2, \dots, D$) and using the time point t as an ending point, a peripheral time region is defined as $[s-0.25d, t]$. In order to be able to uniformly select N time points in the peripheral time region range and sample on a basic feature sequence S_F , a sampling weight matrix $W \in \mathbb{R}^{N \times D}$ is proposed. D values of an i^{th} row $W[i]$ of the sampling weight matrix respectively represent contributing weights of feature vectors of D time points of the feature sequence S_F to an i^{th} sampling time point. For sampling time points as integers, the corresponding column in $W[i]$ is directly assigned as 1 and the others of the row as 0. For sampling time points as non-integers, basic feature vectors corresponding to two integer time points before and after the sampling time point are accumulated by linear interpolation. For example, for the i^{th} sampling time point t_i :

$$W[i, d] = \begin{cases} 1 - \text{dec}(t_i) & \text{if } d = \text{floor}(t_i) \\ \text{dec}(t_i) & \text{if } d = \text{ceil}(t_i) \\ 0 & \text{otherwise} \end{cases} \dots (9)$$

Where $\text{floor}(\cdot)$, $\text{ceil}(\cdot)$, and $\text{dec}(\cdot)$ respectively represent rounding up to the next integer, rounding down to the nearest integer, and setting a decimal.

A boundary matching feature $M \in \mathbb{R}^{C \times N}$ can be obtained by multiplying the feature sequence S_F by the transpose W^T of the sampling weight matrix, and each column corresponds to a sampling time point. For each interval length d between 1 and D , a corresponding sampling weight matrix of $N \times D$ is used to perform sampling on the feature sequence S_F , so the sampling weight matrix is in fact a three-dimensional matrix $W \in \mathbb{R}^{N \times D \times D}$. The feature sequence S_F is multiplied by W^T which is obtained by transposing first two dimensions of the sampling weight matrix, and the result is a complete boundary matching feature matrix $M \in \mathbb{R}^{C \times N \times D}$ that is formulated as:

$$M[c, n, d] = \sum_{i=1}^D S_F[c, i] \times W[n, i, d] \dots (10)$$

A process in which a new feature matrix with a duration N is sampled for one time in the peripheral time regions of all intervals having a length from 1 to D and using t as an ending point is implemented. It should be noted that the value of the sampling weight matrix W is fixed regardless of the value of the feature sequence S_F . Therefore, the value of the sampling weight matrix W can be calculated in advance for one time without real-time calculation during training or testing.

The boundary matching feature matrix M obtained by the boundary matching layer is inputted into a subsequent four-layer convolutional neural network. The two-dimensional convolution layer Conv2d acts on both dimensions N and D . The convolution kernel sizes are 32 and 1, respectively, and information of sampling time points are integrated. Thereafter, the two one-dimensional convolution layers Conv1d 1 and Conv1d 2 both act on one dimension D , and the convolution kernel sizes are 1 and 3, respectively, to integrate information of channels and information of lengths of adjacent proposals. The above three convolutional layers use ReLU as an activation function. The last one-dimensional convolutional layer Conv1d 3 also acts on D , the convolutional kernel size is 1, and two confidence vectors M_{cc} , $M_{cr} \in \mathbb{R}^D$ are outputted by using the Sigmoid activation function. The classification confidence vector M_{cc} denotes the probability of whether the candidate action proposal having a length from 1 to D and using a current time as an ending point is an action proposal and is used for auxiliary classification, and the regression confidence vector M_{cr} denotes the probability that the candidate action proposal having a length from 1 to D and using a current time as an ending point is close to an ground truth action proposal, and is used for accurate prediction. That is, the two confidence vectors outputted by the proposal evaluation processing include information of a possibility that a candidate action proposal having a length from 1 to D and

using a time point t as an end time. When the current time $t < D$, the input of the proposal evaluation processing will add padding whose values are all zero to complete the length D of the feature sequence, and the corresponding output classification confidence vector M_{cc} and regression confidence vector M_{cr} will still have the length D , but only the part with a proposal length from 1 to t is used in calculating the loss. Specifically, the label of the proposal evaluation processing is $G_c \in \mathbb{R}^{D \times T}$, and $G_c[d, t]$ represents the possibility that the time interval having a length d and using t as an ending point is an action proposal. Corresponding to the time point t , the output classification confidence vector M_{cc} and the regression confidence vector M_{cr} correspond to the t^{th} column of G_c , that is, $G_c[:, t]$. The classification confidence vector M_{cc} is used to calculate a weighted cross entropy loss (WCE) with $G_c[:, t]$, so that the positive example approaches 1 and the negative example approaches 0, which helps to correctly classify the positive example and the negative example. The regression confidence vector M_{cr} is used to calculate a mean square error loss (MSE) with $G_c[:, t]$, the purpose is to make the confidence vector as close as possible to an action proposal possibility label. Both confidence vectors are designed to calculate the above two losses, and together represent the probability that the candidate action proposal is an action proposal. FIG. 10 is a diagram of a confidence vector outputted by proposal evaluation processing according to some embodiments of the present application. As shown in FIG. 10, a coordinate of a horizontal axis is a time interval from d to a time point t , and a coordinate of a vertical axis is a probability value from 0 to 1. A solid line indicates a probability of whether a candidate action proposal having a length from 1 to D and using a current time as an ending point is an action proposal, that is, a classification confidence vector M_{cc} , and a dashed line indicates a proportion that a candidate action proposal having a length from 1 to D and using a current time as an ending point is close to a ground truth action proposal, that is, a regression confidence vector M_{cr} .

The candidate action proposal generation processing in the present application is also referred to as an online boundary matching network. The function of the proposal evaluation processing is similar to that of the proposal evaluation processing of the (offline) boundary matching network (BMN), and the function is that a boundary matching feature matrix is obtained by selecting N sampling time points for the feature sequences within and around a series of time intervals, and then a confidence vector (the online boundary matching network) or a confidence matrix (the (offline) boundary matching network) is obtained by a series of convolution operations. The difference is that the (offline) boundary matching network performs proposal evaluation processing on the complete video feature sequence for one time,

and samples any time interval with a length less than D . However, the online boundary matching network pays attention to only the time interval having a length smaller than D and using t as an ending point, so that one dimension is reduced in terms of parameters, operation, and result, that is, the time dimension T . For example, the proposal evaluation processing of the (offline) boundary matching network outputs the confidence matrices $M_{cc}, M_{cr} \in \mathbb{R}^{D \times T}$, while the online boundary matching network outputs the confidence vectors $M_{cc}, M_{cr} \in \mathbb{R}^D$.

The steps 4022 and 4023 may be performed in parallel, and the order herein is merely exemplary.

Step 4024: Calculate a confidence score based on the starting point probability sequence, the ending point probability sequence, the classification confidence vector, and the regression confidence vector.

Specifically, after a t^{th} feature vector is inputted to the online boundary matching network at the time point t , a probability $p_{s,t}$ that the time point t is used as a starting point and a probability $p_{e,t}$ that the time point t is used as an ending point can be obtained from the temporal evaluation processing, and confidence vectors M_{cc} and M_{cr} that t is used as the ending point can be obtained from the proposal evaluation processing.

In order to give the final generated action proposal, first, there is a need to determine whether the time point t is the starting point or the ending point by using the probability. If the starting point probability $p_{s,t}$ meets $p_{s,t} > 0.5 \times \max_{i=0,1,\dots,t-1} (p_{s,i})$ or $p_{s,t} > p_{s,t-1}$, t is regarded as a

candidate action starting point and stored in a list including all candidate action starting points between 0 and t . Whether the time point t is a candidate action ending point is determined by a method similar to that for determining the candidate action starting point. If the time point t does not meet a requirement of the candidate action ending point, no action proposal is given at a current time, and a basic feature vector of a next time is directly awaited to be processed. If the time point t is the candidate action ending point, all the starting points that can form the candidate action proposal having a length smaller than a maximum length D with the time point t are selected from the candidate action starting point list, and all the generated candidate action proposals are passed to the next step. Specifically, D can be set to be 20s.

The confidence score of the proposal is given by comprehensively considering the starting point probability, the ending point probability, and the classification confidence vector and the regression confidence vector. The confidence score $score_i$ of ϕ_i is defined as:

$$score_i = p_{s,t_{s,i}} \times p_{e,t} \times \sqrt{M_{cc,t}[t-t_{s,i}+1] \times M_{cr,t}[t-t_{s,i}+1]} \dots \quad (11)$$

Where $t-t_{s,i}+1$ is a length of the candidate action proposal ϕ_i . The set of candidate action proposals with confidence scores may be represented as $\Phi_c = \{\phi_i = [t_{s,i}, t, score_i]\}_{i=1}^{N_c}$.

In order to verify the effect of the online boundary matching network (Online BMN) method and to compare this method with the offline method and the baseline method, the present application designs and performs a number of sets of experiments to comprehensively examine accuracy and real-time performance of the online boundary matching network. The experiment is based on a large-scale video analysis dataset ActivityNet v1.3 including 27801 videos and applicable to multiple video analysis tasks including action recognition and temporal action detection. Two methods are used to extract features from the videos in the ActivityNet v1.3 dataset in the present application. Because two sets of features obtained by the two methods are different for the temporal action detection method in which the features are directly input, and can be considered as different datasets. For convenience, the feature dataset extracted by using the two-stream convolutional network is denoted as ActivityNet v1.3 TS, and the feature dataset extracted by using the SlowFast network is denoted as ActivityNet v1.3 SF.

A programming language of the code in the experiment is Python 3.7 and a machine learning framework used is PyTorch 1.5. Three temporal action detection methods are implemented:

- 1) An (Offline) Boundary Matching Network (BMN): The portion of the offline boundary matching network that obtains the probabilities that each video segment is a starting point and an ending point is similar to that of a boundary sensitive network, and consists of a convolutional neural network. A boundary matching layer (BM Layer) is used, video feature information of a time period is multiplied by a pre-calculated sampling weight matrix, feature information between all possible pairs of an starting point and an ending point during this time period is sampled as a new feature of a fixed length, and confidence scores of all pairs of an starting point and an ending point are obtained for one time by means of several layers of convolutional neural networks. The generation processing of the candidate action proposal in the online temporal action detection method of the present application is designated as an online boundary matching network (Online BMN), and for better differentiation, the boundary matching network is referred to as an (offline) boundary matching network. In the present application, a model of the (offline) boundary matching network is used to obtain a

comparison effect with the online boundary matching network. During training, the learning rate of the (offline) boundary matching network is 0.001 or 0.0001, the batch size is 64, and each video card is 16. A weight value (functionally identical to that in the online boundary matching network) used to balance a weighted cross entropy loss and a mean square error loss in the proposal evaluation processing is 1.0 or 10.0.

2) An online boundary sensitive network (Online BSN): Probabilities that each video segment is a starting point, an ending point, and a middle point are obtained by passing the input video feature through a one-dimensional convolutional neural network in a three-layer temporal dimension. Then, for each pair of a starting point and an ending point with a high probability, a fixed-length new sequence is sampled from a middle-point probability sequence within an interval from a starting point to an ending point, and the confidence score of the proposal is finally obtained by multiplying the output result by the starting point probability and the ending point probability by a proposal evaluation module (PEM) consisting of two layers of fully connected layers. In the present application, an (offline) boundary sensitive network is changed to an online method as a Baseline. During training, the learning rate of the online boundary sensitive network is 0.0003, a batch size is 128, and each video card is 32.

3) An online boundary matching network (Online BMN): During training, a learning rate of the online boundary matching network is 0.002 or 0.001, a batch size of 128, and each video card is 32. A weight value used to balance a weighted cross entropy loss and a mean square error loss in the proposal evaluation processing is 1.0 or 10.0.

In the above three methods, the maximum proposal length D of the online and offline boundary matching networks has the same value as the feature sequence length T , so as to ensure that the true action proposals of all lengths are covered. The online boundary sensitive network has no limit on the maximum proposal length. The number N of sampling time points of the boundary matching layer of the online and offline boundary matching networks is set to be 32. In addition, the above three methods all train 10 epochs, and an optimizer used adopts an Adam method. Except that the learning rate is determined according to the model, and the weight decay is set to be 0.0001 for regularization, the other parameters use default parameters ($\beta=(0.9, 0.999)$, $\epsilon=1e-8$) of the Adam method in PyTorch 1.5. For parameters with multiple values, experiments are performed on all parameter combinations and the best experimental result is retained.

In an initial experiment, it is found that it takes a long time to or filter out the redundant candidate action proposals in a soft non-maximum suppression method. Although the soft

non-maximum suppression method is simple in process, each time a proposal ϕ_p is selected, there is a need to calculate the intersection over union of the proposal ϕ_p and all the proposals in Φ_c , and update a confidence score accordingly. The time complexity is $O(N_P \cdot |\Phi_c|)$, N_P is the target number of generated action proposals, and $|\Phi_c|$ is the number of proposals in the set of candidate action proposals. When there are thousands of candidate action proposals in the set Φ_c , the soft non-maximum suppression method takes up most of the time of the inference process. To reduce the duration consumed, a source file including the soft non-maximum suppression method is compiled by using Cython, and the source file is replaced with a generated dynamic link library. The experimental results show that efficiency of the soft non-maximum suppression method is improved by 3.5–4 times.

1) Offline detection experiment

In temporal action detection, an offline method can observe a complete video feature sequence, while an online method can only observe, at each time point t at which inference is made, a feature sequence prior to t , and thus the offline method that is structurally similar in theory should be an upper bound of the effect that can be obtained by the online method. The technical solution of the present invention is the online temporal action detection method. However, in order to be able to directly compare with an offline method and observe an effect difference, an offline detection experiment is also provided. In an inference process of the offline detection experiment, only candidate action proposal generation and score fusion are performed, and proposal filtering is not performed at present. When the video processing is completed and sets $\Phi_{c,t}$ of candidate action proposals at time points t have been obtained, these sets are combined into a set of candidate action proposals for the entire video, which is denoted as Φ_c . Finally, Φ_p is obtained by performing proposal filtering for one time for Φ_c by using the soft non-maximum suppression method, and includes at most N_P action proposals. An offline detection experiment is characterized in that all candidate action proposals can be combined and at most N_P action proposals with the highest global confidence score can be selected. The detection method is the same as the detection method of the offline temporal action detection research work (including the offline boundary sensitive network and the offline boundary matching network) in recent years.

A basic evaluation index of a temporal action detection task is a recall rate of a ground truth action proposal, that is, for each ground truth action proposal ϕ_g , a probability that

there is a proposal ϕ_p in the inferred set Φ_p of action proposals matching the ground truth action proposal ϕ_g , and a quantization definition of the matching is that the intersection over union of ϕ_g and ϕ_p reaches a certain set threshold z . On the other hand, the number of action proposals given also affects the recall rate. Therefore, under a condition that the maximum number N_p of generated action proposals is fixed for each video, an average recall rate of ground truth action proposals in each video needs to be measured, and is called AR@AN, that is, the average recall rate under the average action proposal number. In accordance with the general evaluation criteria of the research work on temporal motion detection in recent years, in an offline detection experiment for two datasets ActivityNet v1.3 TS and SF, we consider AR@AN when the intersection over union threshold $z=[0.5:0.05:0.95]$, and AR@AN (denoted as AR@10, AR@50, AR@100, respectively) for three values of $N_p \in \{10, 50, 100\}$. The area under curve (Area Under Curve, AUC) of AR@AN when the values are integers between 1 and 100 is defined as:

$$AUC = \sum_{N_p=1}^{100} AR @ N_p \dots (12)$$

The results of offline detection experiments on the ActivityNet v1.3 TS dataset are shown in Table 4. Generally, the boundary matching network (BMN) is an offline method, and can obtain all information of the feature sequence during the operation, so that the effect is superior to two online methods. In the online method, the online boundary matching network (Online BMN) uses a long short-term memory network that can make better use of history information in the temporal evaluation process, and directly analyzes the feature sequence in the proposal evaluation process, thereby being more adaptable to the temporal action detection task, and the effect of the online BMN is generally superior to that of the online boundary sensitive network (Online BSN) that is used as a baseline. It is worth noting that, when the number N_p of generated action proposals is controlled to be a smaller value of 10, the AR@10 result of the online boundary matching network is nearly 2% higher than that of the online boundary sensitive network, the advantage is obvious, and it is indicated that the online boundary matching network gives a more accurate confidence score for the action proposals, and the candidate action proposals with a higher degree of matching with ground truth action proposals are ranked in the forefront. On the other hand, although there is still a certain difference between the online boundary matching network and the (offline) boundary matching network, the online boundary matching network can achieve an effect of having a difference of only 3%-4% from the (offline) boundary matching network by using only

one-tenth of the parameters of the (offline) boundary matching network. The online boundary matching network still has certain advantages in online temporal action detection tasks that require high real-time performance and tend to be lightweight.

Results of offline detection experiments on the ActivityNet v1.3 SF dataset are shown in Table 5. A trend of the results is similar to results on the ActivityNet v1.3 TS dataset. The increase in the number of parameters is due to the fact that the number of feature channels in the ActivityNet v1.3 SF dataset is much larger than that in the TS dataset.

Table 4

		AR@AN (%)			AUC	Parameters
Model		AR@10	AR@50	AR@100		
BMN	TS	53.29	67.88	73.55	65.26	3.48×10^7
Online BMN	TS	49.43	63.75	70.08	61.46	3.07×10^6
Online BSN	TS	47.51	63.68	70.25	60.81	1.42×10^6

Table 5

		AR@AN (%)			AUC	Parameters
Model		AR@10	AR@50	AR@100		
BMN		54.04	67.44	73.24	64.99	7.14×10^7
Online BMN		48.64	63.89	70.01	61.30	4.74×10^6
Online BSN		47.69	62.56	69.35	59.90	4.35×10^6

2) Online detection experiment

In the offline experiment, any number of time intervals can be put into the set Φ_c of candidate action proposals at a time point t , and proposals with high confidence scores can be filtered out as a whole after a video ends. Different from offline detection experiments, online detection experiments are intended to be as close to a real-time video environment as possible, and give action proposals along with the incoming video. Therefore, when the time point t is reached, it is necessary to select a small number of intervals most likely to match the actual action proposal from the time intervals ending with t according to the circumstances.

To meet the above requirements, in the inference process of the online detection experiment, three stages of candidate action proposal generation, score fusion, and proposal filtering are performed at each time point, and the stage of proposal filtering allows only a

maximum of 10 action proposals to be given at each time point t . However, reducing a redundant action proposal is also one of the targets of the online temporal action detection method. In the method, aiming to achieving a higher recall, unnecessary (low-confidence score or redundant) action proposals may be generated as many as possible at each time point t within a limit of 10 proposals. In order to avoid this case, at the end of the method, at most 10 action proposals generated at each time point are put into a set Φ_c and then subjected to a proposal filtering by using a soft non-maximum suppression method, and no more than N_p action proposals are finally filtered out. Ideally, if the method evaluates the confidence score for the time interval sufficiently well, the final proposal filtering may not be performed in an application scenario, and only the confidence score threshold is set as an output criterion, so that a proper number of action proposals can be outputted according to circumstances in the method, and it can be ensured that the inference process of the method is performed entirely online.

The evaluation criterion of the online detection experiment is the same as that of the offline detection, and is $AR@AN$ when the intersection over union threshold $z=[0.5:0.05:0.95]$. $AR@AN$ (denoted as $AR@10$, $AR@50$, $AR@100$, respectively) for three values of $N_p \in \{10, 50, 100\}$, and the area AUC under curve of $AR@AN$ when the values are integers between 1 and 100 are considered.

The (offline) boundary matching network (BMN) cannot perform online detection, so we only compare the results of the online boundary matching network (Online BMN) with that of the online boundary sensitive network (Online BSN) as a baseline. It can be learned from Table 6 that the effect of the online boundary matching network is generally superior to that of the online boundary sensitive network in the online detection experiment, and the online experiment has a more significant improvement than the offline experiment, indicating that the online boundary matching network can better learn the video features in the online environment. Similar to a case in the offline detection experiment, when the total number of generated action proposals is controlled to be a smaller value of 10, the online boundary matching network has an improvement of more than 2% on both the ActivityNet v1.3 TS and SF datasets compared with the online boundary sensitive network, and accuracy of the online boundary matching network in evaluating the confidence scores of the action proposals is demonstrated again.

Table 6

Model		AR@AN (%)			AUC	Single-step Time (ms)
		AR@10	AR@50	AR@100		
Online BMN	TS	50.25	64.42	70.34	62.00	2.3
Online BSN	TS	48.11	64.14	69.80	61.07	12.1
Online BMN	SF	50.13	64.26	69.77	61.88	3.2
Online BSN	SF	47.82	63.35	68.84	60.32	14.3

On the other hand, data of a single-step time column represents the time from receiving the input of the feature vector to giving no more than 10 candidate action proposals ending at the current time when the feature vectors of a video feature sequence are sequentially processed by the method. This part of the test is done only on a CPU without using a GPU. The data shows that a processing speed of the online boundary matching network is four to five times more than that of the online boundary sensitive network, and real-time performance is higher. Even on a platform with a limited computational capability, or video features need to be extracted simultaneously when end-to-end online temporal action detection is used, a more efficient online boundary matching network method can maintain a finer granularity in a time dimension.

3) Structure comparison experiment

In the temporal action detection method proposed in recent years, modules similar to a function of the temporal evaluation processing of the online boundary matching network generally use a convolutional neural network to obtain a start point probability sequence and an ending point probability sequence. An advantage of using the convolutional neural network is that the convolutional neural network has a strong parallelism and runs fast in an offline temporal action detection method. However, the convolutional neural network also has disadvantages of having a limited receptive field and not being able to make good use of history information of a feature sequence, and this is particularly important in an online temporal action detection in which features after a current time cannot be known. In order to verify the improvement of the effect of replacing the commonly used convolutional neural network with the long short-term memory network (LSTM) as the main structure of the temporal evaluation processing, a structure comparison experiment of the online boundary matching network with two kinds of temporal evaluation processing on the ActivityNet v1.3 TS dataset is designed. The proposed online boundary matching network uses a long short-term memory network, which is temporarily denoted as Online BMN-RNN in the structure comparison experiment. A model using a two-layer online one-dimensional convolutional neural network is denoted as Online BMN-CNN, and a structure of a temporal

evaluation module TEM is shown in Table 7. The experimental results shown in Table 8 and Table 9 show that Online BMN-RNN achieves better effects than Online BMN-CNN no matter whether in the offline detection experiment or in the online detection experiment. In addition, although the long short-term memory network has more parameters and more complex operations than the online one-dimensional convolution, the single-step operation time of the Online BMN-RNN is still 0.3 milliseconds faster than that of the Online BMN-CNN.

Table 7

Layer	Kernel Size	Output Channel	Activation	Output Size
Online Conv1d 1	3	256	ReLU	256
Online Conv1d 2	3	2	ReLU	2

10

Table 8

Model	AR@AN (%)			AUC	Parameters
	AR@10	AR@50	AR@100		
Online BMN-RNN	49.43	63.75	70.08	61.46	3.07×10^6
Online BMN-CNN	47.66	63.30	69.89	60.62	3.04×10^6

Table 9

Model	AR@AN (%)			AUC	Single-step Time (ms)
	AR@10	AR@50	AR@100		
Online BMN-RNN	50.25	64.42	70.34	62.00	2.3
Online BMN-CNN	48.04	63.86	69.90	61.02	2.6

In the online temporal action detection method of the present application, operations such as a lightweight online one-dimensional convolution, a long short-term memory network, and a boundary matching layer are used, so that a probability that a current time point is used as a proposal boundary can be inferred along with real-time incoming video streams, and possibility that each time period using the current time point as an ending point is used as a proposal interval can be investigated. Experimental results of offline detection and online detection respectively performed on two datasets: ActivityNet v1.3 TS and SF show that the online boundary matching network (Online BMN) can achieve better effects than a baseline, that is, the online boundary sensitive network (Online BSN). Compared with the prior art, the online temporal action detection method of the present invention is more accurate in estimating confidence scores of candidate action proposals. In an actual application scenario of the online temporal action detection, in order to control redundancy, when too many action proposals cannot be generated at each time point t , a higher recall rate can be achieved with fewer proposals. On the other hand, the online boundary matching network has less than one tenth of the number of parameters compared with the (offline) boundary matching network (BMN) of a similar structure. In a case of a high requirement for real-time performance, a millisecond-level single-step calculation time can also be reached without relying on an accelerated operation of a video card, and is four to five times faster than that of the baseline that is the online boundary sensitive network. Even in an end-to-end environment in which a computing capability of a platform is limited and it is necessary to extract features from the generated video stream in real time and then perform action detection, the online action detection method may use an application scenario requires a finer granularity in a time

dimension.

According to some embodiments of the present application, an online temporal action detection apparatus 600 is provided. FIG. 11 is a schematic structural diagram of an online temporal action detection apparatus according to some embodiments of the present application. As shown in FIG. 11, the apparatus 600 for performing online temporal action detection is as follows:

a video obtaining unit 601, configured to obtain a video online and extract an initial feature vector for action detection from the video;

a candidate action proposal generation unit 602, configured to construct a candidate action proposal based on the initial feature vector, and generate a confidence score associated with the candidate action proposal based on the candidate action proposal and a corresponding ground truth action proposal in the video, where the confidence score represents a degree of matching between the candidate action proposal and the ground truth action proposal in the video;

an action proposal filtering unit 603, configured to filter out a redundant action proposal from the candidate action proposal and output a generated action proposal;

where the candidate action proposal generating unit 602 includes:

an online one-dimensional convolution operation subunit 6021, configured to perform an online one-dimensional convolution operation for at least one time based on the initial feature vector to output a basic feature vector one time point by one time point; and

respectively input the basic feature vector into a temporal evaluation subunit 6022 and a proposal evaluation subunit 6023 to perform an operation, where

the temporal evaluation subunit 6022 is configured to receive a single basic feature vector one time point by one time point, and perform an operation including a recurrent neural network to generate a starting point probability sequence and an ending point probability sequence of the candidate action proposal one time point by one time point;

the proposal evaluation subunit 6023 is configured to cumulatively receive a predetermined number of basic feature vectors generated one time point by one time point, perform an operation including sampling and convolution, and output a classification confidence vector and a regression confidence vector; and

a confidence score calculation subunit 6024, configured to calculate the confidence score based on the starting point probability sequence, the ending point probability sequence, the classification confidence vector, and the regression confidence vector.

In some embodiments, in the online one-dimensional convolution operation subunit 6022, the initial feature vector is inputted to a first feature buffer one time point by one time point, where the first feature buffer can store k initial feature vectors arranged one time point by one time point, and k is a size of a convolution kernel of the online one-dimensional convolution operation.

In some embodiments, the online one-dimensional convolution operation subunit 6021 further includes:

the first feature buffer, used to receive the initial feature vector one time point by one time point and delete the initial feature vector first inputted into the first feature buffer, to ensure that k initial feature vectors most recently inputted one time point by one time point are stored in the first feature buffer, where k represents the size of the convolution kernel; and

a one-dimensional convolution layer, used to perform a convolution operation on the k initial feature vectors in the first feature buffer, and obtain the basic feature vector as an output result.

In some embodiments, the temporal evaluation subunit 6022 includes:

receiving a single basic feature vector one time point by one time point and performing an operation including a recurrent neural network, wherein processing on the basic feature vector at a current time point by the recurrent neural network combines a result of processing on the basic feature vector at a previous time point.

In some embodiments, the temporal evaluation subunit 6022 further includes:

a long short-term memory network, the recurrent neural network being a long short-term memory network, used to receive a single basic feature vector, an original hidden state h_{t-1} and an original cell state c_{t-1} obtained from a long short-term memory network of a previous time point, and calculate and output a current new hidden state h_t based on the basic feature vector, the original hidden state h_{t-1} and the original cell state c_{t-1} ;

a fully connected layer, used to receive the new hidden state h_t one time point by one time point, and output the starting point probability sequence and the ending point probability sequence by using an activation function, where the starting point probability consists of a probability $p_{s,t}$ that a time point t is used as a starting point of the candidate action proposal, and the ending point probability consists of a probability $p_{e,t}$ that a time point t is used as an ending point of the candidate action proposal.

In some embodiments, the apparatus 600 further includes setting an initial hidden state and an initial cell state to be 0.

In some embodiments, in the proposal evaluation subunit 6023, the basic feature vector

is inputted into a second feature buffer one time point by one time point, where the second feature buffer cumulatively receives and stores a predetermined number of basic feature vectors generated one time point by one time point.

In some embodiments, the proposal evaluation subunit 6023 further includes:

5 the second feature buffer, used to accumulatively receive a predetermined number of basic feature vectors generated one time point by one time point and store the basic feature vectors as a basic feature sequence $S_F \in \mathbb{R}^{C \times D}$;

a boundary matching layer, used to perform point multiplication on the basic feature sequence and a sampling weight matrix $W \in \mathbb{R}^{N \times D}$, and obtain and output a boundary matching matrix $M \in \mathbb{R}^{C \times N}$ having a sampling duration N and a time region length from 1 to D , and using a time point t as an ending point; and

10 a convolutional neural network, the convolutional neural network comprising at least one layer, used to input the boundary matching matrix to perform a convolution operation, and output the classification confidence vector $M_{cc} \in \mathbb{R}^D$ and the regression confidence vector $M_{cr} \in \mathbb{R}^D$;

15 where C denotes the number of channels of the basic feature vector, D denotes a length of the longest candidate action proposal in the basic feature sequence, the sampling weight matrix denotes a weight of the basic feature sequence S_F in the sampling duration N , M_{cc} denotes a probability of whether the candidate action proposal having a length from 1 to D and using a current time as an ending point is an action proposal, and M_{cr} denotes a probability that the candidate action proposal having a length from 1 to D and using a current time as an ending point is close to an ground truth action proposal.

In some embodiments, the at least one layer of convolutional neural network may include four layers of convolutional layers, where

25 the first layer is a two-dimensional convolutional layer, and acts on two dimensions of the sampling duration N and the length D of the boundary matching matrix to integrate information of sampling points;

both the second layer and the fourth layer are one-dimensional convolution layers, and act on a dimension of the length D to integrate information of channels; and

30 the third layer is a one-dimensional convolutional layer, and acts on a dimension of the length D to integrate information of lengths of adjacent proposals.

In some embodiments, the constructing a candidate action proposal further includes:

determining that a time point t is a candidate action starting point, and adding the time

point t of the candidate action starting point to a list of candidate action starting points;

determining that a time point t is a candidate action ending point, selecting, from the list of candidate action starting points, the candidate starting point that can form the candidate action proposal having a length smaller than a maximum length D with the time point t , and

5 outputting a set of candidate action proposals $\Phi_c = \{\phi_i = [t_{s,i}, t]\}_{i=1}^{N_c}$;

where ϕ_i denotes an i^{th} candidate action proposal, N_c denotes the number of candidate action proposals, and $t_{s,i}$ denotes a start time of the i^{th} candidate action proposal.

In some embodiments, the confidence score of the candidate action proposal is calculated by the following formula:

$$10 \quad \text{score}_i = p_{s,t_{s,i}} \times p_{e,t} \times \sqrt{M_{cc,t}[t - t_{s,i} + 1] \times M_{cr,t}[t - t_{s,i} + 1]},$$

where $t - t_{s,i} + 1$ denotes a length of the candidate action proposal ϕ_i , $p_{s,t}$ and $p_{e,t}$ respectively denote probabilities that a time point t is used as a starting point and an ending point of the candidate action proposal, $M_{cc,t}$ denotes probabilities that the candidate action proposal having a length from 1 to D and using a time point t as an ending point is an action
15 proposal, and $M_{cr,t}$ denotes a probability that the candidate action proposal having a length from 1 to D and using a time point t as an ending point is close to the ground truth action proposal.

In some embodiments, the initial feature vector includes temporal structure information.

In the online temporal action detection apparatus of the present application, operations
20 such as a lightweight online one-dimensional convolution, a long short-term memory network, and a boundary matching layer are used, so that a probability that a current time point is used as a proposal boundary can be inferred along with real-time incoming video streams, and possibility that each time period using the current time point as an ending point is used as a proposal interval can be investigated. Compared with the prior art, the online temporal action
25 detection apparatus of the present application is more accurate in estimating a confidence score of a candidate action proposal. In an actual application scenario of online temporal action detection, in order to control redundancy, when too many action proposals cannot be generated at each time point t , a higher recall rate can be achieved with fewer proposals. On the other hand, an online boundary matching network has less than one tenth of the number of
30 parameters compared with an (offline) boundary matching network (BMN) with a similar structure. In a case of a high requirement for real-time performance, a millisecond-level single-step calculation time can also be achieved without relying on an accelerated operation

of a video card, and is four to five times faster than that of a baseline that is an online boundary sensitive network. Even in an end-to-end environment in which a computing capability of a platform is limited and it is necessary to extract features from the generated video stream in real time and then perform action detection, the online action detection apparatus may use an application scenario requires a finer granularity in a time dimension.

It can be understood that the image-based human body detection device corresponds to the image-based human body detection method provided by the present application, and technical details in the above specific description about the image-based human body detection method provided by the present application are still applicable to the image-based human body detection device. For a specific description, please refer to the above description. Details are not repeated herein.

According to some embodiments of the present application, a computer readable storage medium encoded with a computer program is provided. It can be understood that the computer readable storage medium encoded with the computer program corresponds to the online temporal action detection method provided by the present application, and the technical details in the above specific description about the online temporal action detection method provided by the present application are still applicable to the computer readable storage medium encoded with a computer program. For a specific description, please refer to the above description. Details are not repeated herein.

It can be understood that exemplary applications of online temporal action detection provided by the embodiments of the present application include but are not limited to image generation in the field of artificial intelligence.

It can be understood that, as used in the description, the terms "module" and "unit" may refer to or include an application specific integrated circuit (ASIC), an electronic circuit, a processor that executes one or more software or firmware programs (shared, dedicated, or group) and/or a memory, a combinational logic circuit, and/or another suitable hardware component that provides the described function, or may be a part of these hardware components.

It can be understood that in the embodiments of the present application, the processor may be a microprocessor, a digital signal processor, a microcontroller, etc., and/or any combination thereof. According to another aspect, the processor may be a single-core processor, a multi-core processor, etc., and/or any combination thereof.

It can be understood that the online temporal action detection method provided in the present application can be implemented on various electronic devices, including but not

limited to electronic devices such as servers, distributed server clusters including a plurality of servers, mobile phones, tablet computers, laptop computers, desktop computers, wearable devices, head-mounted displays, mobile email devices, portable game consoles, portable music players, reader devices, personal digital assistants, virtual reality or augmented reality devices, and TV sets with one or more embedded or coupled processors.

In particular, the online temporal action detection method provided by the present application is suitable for an edge device. Edge computing adjacently provides edge intelligence services by using a distributed open platform (architecture) that integrates core capabilities of network, computing, storage, and application on the edge of the network near the source of objects or data, so that key requirements of real-time service, data optimization, application intelligence, security and privacy protection can be met. For example, the edge device may be a device capable of performing edge calculation on image data in a video at an end close to a video data source (a network intelligent camera) in a video surveillance system.

All the embodiments of the mechanism disclosed in the present application may be implemented in hardware, software, firmware, or a combination of these implementation methods. The embodiments of the present application may be implemented as computer programs executed on a programmable system or program codes, and the programmable system includes at least one processor, a storage system (including a volatile and non-volatile memory and/or a storage element), at least one input device, and at least one output device.

The program codes can be applied to input an instruction to perform the functions described in the present application and generate output information. The output information can be applied to the one or more output devices in a known manner. For the objective of the present application, a processing system includes any system having a processor such as a digital signal processor (DSP), a microcontroller, an application-specific integrated circuit (ASIC), or a microprocessor.

The program codes can be implemented in a high-level programming language or an object-oriented programming language in order to communicate with the processing system. When needed, the program codes can also be implemented in an assembly language or a machine language. In fact, the mechanism described in the present application is not limited to the scope of any particular programming language. In either case, the language can be a compiled language or an interpreted language.

In some cases, the disclosed embodiments may be implemented in hardware, firmware, software, or any combination thereof. The disclosed embodiments may further be

implemented as instructions carried on or stored by one or more transitory or non-transitory machine-readable (such as computer readable) storage media, and the instructions may be read or executed by one or more processors. For example, the instructions may be distributed by using a network or by using another computer readable medium. Therefore, the machine-readable medium may include any mechanism for storing or transmitting information in a machine (such as a computer) readable form, including but not limited to, a floppy disk, an optical disk, an optical disc, a read-only memory (CD-ROM), a magneto-optical disk, a read-only memory (ROM), a random access memory (RAM), an erasable programmable read-only memory (EPROM), an electrically erasable programmable read-only memory (EEPROM), a magnetic card or an optical card, a flash memory, or a tangible machine-readable memory used to transmit information through propagation signals (such as a carrier signal, an infrared signal and a digital signal) in electricity, light, sound or other forms through the Internet. Therefore, the machine-readable medium includes any type of machine-readable medium suitable for storing or transmitting an electronic instruction or information in a machine (such as the computer) readable form.

In the drawings, some structural or method features may be shown in a specific arrangement and/or order. However, it should be understood that such a specific arrangement and/or ordering may not be required. Rather, in some embodiments, these features may be arranged in a manner and/or order different from that shown in the illustrative drawings. In addition, the inclusion of structural or method features in a particular figure does not imply that such features are required in all embodiments, and in some embodiments, these features may not be included or may be combined with other features.

It should be noted that the units/modules mentioned in the device embodiments of the present application are logical units/modules. Physically, a logical unit/module may be a physical unit/module or may be a part of a physical unit/ module, or may be implemented as a combination of multiple physical units/modules. The physical implementation of these logical units/modules is not the most important. The combination of the functions implemented by these logical units/modules is the key to solving the technical problem presented in the

present application. In addition, in order to highlight the innovation part of the present application, the above device embodiments of the present application do not introduce units/modules that are not closely related to solving the technical problems raised by the present application, and this does not mean that there are no other units/modules in the above
5 device embodiments.

It should be noted that in the examples and the description of the present application, relational terms such as "first" and "second" are only used to distinguish one entity or operation from another entity or operation, and do not necessarily require or imply that there is any such actual relationship or order between these entities or operations. Moreover, the
10 terms "comprise", "include" or any other variants thereof are intended to cover the non-exclusive inclusion, so that a process, method, system, product, or device that includes a list of elements not only includes those elements but also includes other elements that are not expressly listed, or further includes elements inherent to such a process, method, article, or device. An element defined by "includes a ..." does not, without more constraints, preclude the
15 existence of additional identical elements in the process, method, product, or device that includes the element.

Although the present application has been illustrated and described by reference to some preferred embodiments of the present application, a person of ordinary skill in the art should understand that various changes can be made in form and detail without departing from the
20 spirit and scope of the present application.

Claims

1. An online temporal action detection method, comprising:

obtaining a video online and extracting an initial feature vector for action detection from the video;

performing candidate action proposal generation processing, constructing a candidate action proposal according to the initial feature vector, and generating a confidence score associated with the candidate action proposal, wherein the confidence score represents a degree of matching between the candidate action proposal and the ground truth action proposal in the video; and

performing action proposal filtering processing, to filter out a redundant action proposal from the candidate action proposal, and output a generated action proposal;

wherein in the performing the generation processing on a candidate action proposal:

performing an online one-dimensional convolution operation for at least once based on the initial feature vector, and outputting a basic feature vector one time point by one time point; and

respectively performing temporal evaluation processing and proposal evaluation processing based on the basic feature vector, wherein

in the temporal evaluation processing, receiving a single basic feature vector one time point by one time point, and performing an operation comprising a recurrent neural network, to generate a starting point probability sequence and an ending point probability sequence of the candidate action proposal one time point by one time point;

in the proposal evaluation processing, cumulatively receiving a predetermined number of basic feature vectors generated one time point by one time point, performing an operation comprising sampling and convolution, and outputting a classification confidence vector and a regression confidence vector; and

calculating the confidence score based on the starting point probability sequence, the ending point probability sequence, the classification confidence vector, and the regression confidence vector.

2. The method according to claim 1, wherein in the online one-dimensional convolution operation, inputting the initial feature vector to a first feature buffer one time point by one time point, wherein the first feature buffer can store k initial feature vectors arranged one time point by one time point, and k is a size of a convolution kernel of the online one-dimensional convolution operation.

3. The method according to claim 2, wherein inputting the initial feature vector to a first

feature buffer one time point by one time point further comprises,

inputting the initial feature vector into the first feature buffer in online one-dimensional convolution one time point by one time point, and deleting the initial feature vector first inputted into the first feature buffer, to ensure that k initial feature vectors inputted most recently one time point by one time point are stored in the first feature buffer, wherein k represents the size of the convolution kernel; and

the online one-dimensional convolution operation further comprises,

performing a convolution operation on the k initial feature vectors in the first feature buffer by using the convolutional kernel, and obtaining the basic feature vector as an output result.

4. The method according to claim 1, wherein the temporal evaluation processing comprises:

receiving a single basic feature vector one time point by one time point, and performing an operation comprising a recurrent neural network, wherein processing on the basic feature vector at a current time point by the recurrent neural network combines a result of processing on the basic feature vector at a previous time point.

5. The method according to claim 4, wherein the recurrent neural network further comprises,

the recurrent neural network is a long short-term memory network;

receiving, by the long short-term memory network, a single basic feature vector, an original hidden state h_{t-1} and an original cell state c_{t-1} obtained from a long short-term memory network of a previous time point, and calculating and outputting a current new hidden state h_t based on the basic feature vector, the original hidden state h_{t-1} and the original cell state c_{t-1} ; and

the temporal evaluation processing further comprises,

inputting the new hidden state h_t into a fully connected layer one time point by one time point, and outputting the starting point probability sequence and the ending point probability sequence by using an activation function, wherein the starting point probability sequence consists of a probability $p_{s,t}$ that a time point t is used as a starting point of the candidate action proposal, and the ending point probability sequence consists of a probability $p_{e,t}$ that the time point t is used as an ending point of the candidate action proposal.

6. The method according to claim 5, wherein the method further comprises setting an initial hidden state h_0 and an initial cell state c_0 to be 0.

7. The method according to claim 1, wherein in the proposal evaluation processing, inputting the basic feature vector to a second feature buffer one time point by one time point, wherein the second feature buffer cumulatively receives and stores a predetermined number of basic feature vectors one time point by one time point.

8. The method according to claim 7, wherein inputting the basic feature vector to a second feature buffer one time point by one time point further comprises:

accumulatively receiving, by the second feature buffer, the predetermined number of basic feature vectors generated one time point by one time point, and storing the basic feature vectors as a basic feature sequence $S_F \in \mathbb{R}^{C \times D}$;

the proposal evaluation processing further comprises,

inputting the basic feature sequence and a sampling weight matrix $W \in \mathbb{R}^{N \times D}$ to a boundary matching layer to perform point multiplication, and obtaining and outputting a boundary matching matrix $M \in \mathbb{R}^{C \times N}$ having a sampling duration N and a time region length from 1 to D , and using the time point t as an ending point; and

inputting the boundary matching matrix into a convolutional neural network including at least one layer to perform convolution operation, and outputting the classification confidence vector $M_{cc} \in \mathbb{R}^D$ and the regression confidence vector $M_{cr} \in \mathbb{R}^D$;

wherein C denotes the number of channels of the basic feature vector, D denotes the length of the longest candidate action proposal in the basic feature sequence, the sampling weight matrix denotes the weight of the basic feature sequence S_F in the sampling duration N , M_{cc} denotes the probability of whether the candidate action proposal having a length from 1 to D and using a current time as an ending point is an action proposal, and M_{cr} denotes the probability that the candidate action proposal having a length from 1 to D and using a current time as an ending point is close to an ground truth action proposal.

9. The method according to claim 8, wherein the at least one layer of convolutional neural network may comprise four layers of convolutional layers, wherein

the first layer is a two-dimensional convolutional layer, and acts on two dimensions of the sampling duration N and the length D of the boundary matching matrix to integrate information of sampling points;

both the second layer and the fourth layer are one-dimensional convolution layers, and act on a dimension of the length D to integrate information of channels; and

the third layer is a one-dimensional convolutional layer, and acts on a dimension of the length D to integrate information of lengths of adjacent proposals.

10. The method according to claim 1, wherein the constructing a candidate action proposal further comprises:

determining that the time point t is a candidate action starting point, and adding the time point t of the candidate action starting point to a list of candidate action starting points;

determining that the time point t is a candidate action ending point, selecting, from the list of candidate action starting points, the candidate starting point that can form the candidate action proposal having a length smaller than a maximum length D with the time point t , and outputting a set of candidate action proposals $\Phi_c = \{\phi_i = [t_{s,i}, t]\}_{i=1}^{N_c}$;

wherein ϕ_i denotes an i^{th} candidate action proposal, N_c denotes the number of candidate action proposals, and $t_{s,i}$ denotes a start time of the i^{th} candidate action proposal.

11. The method according to claim 1, wherein the confidence score is calculated by the following formula:

$$\text{score}_i = p_{s,t_{s,i}} \times p_{e,t} \times \sqrt{M_{cc,t}[t - t_{s,i} + 1] \times M_{cr,t}[t - t_{s,i} + 1]},$$

wherein $t - t_{s,i} + 1$ denotes a length of the candidate action proposal ϕ_i , $p_{s,t}$ and $p_{e,t}$ respectively denote probabilities that the time point t is used as a starting point and an ending point of the candidate action proposal, $M_{cc,t}$ denotes the probability of whether the candidate action proposal having a length from 1 to D and using a time point t as an ending point is an action proposal, and $M_{cr,t}$ denotes the probability that the candidate action proposal having a length from 1 to D and using a time point t as an ending point is close to the ground truth action proposal.

12. An online temporal action detection apparatus, comprising:

a video obtaining unit, configured to obtain a video online and extracting an initial feature vector for action detection from the video;

a candidate action proposal generation unit, configured to construct a candidate action proposal based on the initial feature vector, and generate a confidence score associated with the candidate action proposal, wherein the confidence score represents a degree of matching between the candidate action proposal and the ground truth action proposal in the video; and

an action proposal filtering unit, configured to filter out a redundant action proposal from the candidate action proposal and output a generated action proposal;

wherein in the candidate action proposal generating unit:

an online one-dimensional convolution operation subunit is configured to perform an online one-dimensional convolution operation for at least once based on the initial feature vector, and output a basic feature vector one time point by one time point; and

respectively input the basic feature vector into a temporal evaluation subunit and a proposal evaluation subunit to perform an operation, wherein

the temporal evaluation subunit is configured to receive a single basic feature vector one

time point by one time point, and perform an operation comprising a recurrent neural network, to generate a starting point probability sequence and an ending point probability sequence of the candidate action proposal one time point by one time point;

the proposal evaluation subunit is configured to cumulatively receive a predetermined number of basic feature vectors generated one time point by one time point, perform an operation comprising sampling and convolution, and output a classification confidence vector and a regression confidence vector; and

a confidence score calculation subunit is configured to calculate the confidence score based on the starting point probability sequence, the ending point probability sequence, the classification confidence vector, and the regression confidence vector.

13. The apparatus according to claim 12, wherein in the online one-dimensional convolution operation subunit, inputting the initial feature vector to a first feature buffer one time point by one time point, wherein the first feature buffer can store k initial feature vectors arranged one time point by one time point, and k is a size of a convolution kernel of the online one-dimensional convolution operation.

14. The apparatus according to claim 13, wherein the online one-dimensional convolution operation subunit further comprises:

the first feature buffer, used to receive the initial feature vector one time point by one time point and delete the initial feature vector first inputted into the first feature buffer, to ensure that k initial feature vectors most recently inputted one time point by one time point are stored in the first feature buffer, wherein k represents the size of the convolution kernel; and

a one-dimensional convolution layer, used to perform a convolution operation on the k initial feature vectors in the first feature buffer, and obtain the basic feature vector as an output result.

15. The apparatus according to claim 12, wherein the temporal evaluation subunit comprises:

receiving a single basic feature vector one time point by one time point and performing an operation comprising a recurrent neural network, wherein processing on the basic feature vector at a current time point by the recurrent neural network combines a result of processing on the basic feature vector at a previous time point.

16. The method according to claim 15, wherein the temporal evaluation subunit further comprises:

a long short-term memory network, the recurrent neural network being a long short-term memory network, used to receive a single basic feature vector, an original hidden state h_{t-1} and an

original cell state c_{t-1} obtained from a long short-term memory network of a previous time point, and calculating and outputting a current new hidden state h_t based on the basic feature vector, the original hidden state h_{t-1} and the original cell state c_{t-1} ; and

a fully connected layer, used to receive the new hidden state h_t one time point by one time point, and output the starting point probability sequence and the ending point probability sequence by using an activation function, wherein the starting point probability sequence consists of a probability $p_{s,t}$ that a time point t is used as a starting point of the candidate action proposal, and the ending point probability sequence consists of a probability $p_{e,t}$ that the time point t is used as an ending point of the candidate action proposal.

17. The apparatus according to claim 12, wherein in the proposal evaluation subunit, inputting the basic feature vector into a second feature buffer one time point by one time point, wherein the second feature buffer cumulatively receives and stores a predetermined number of basic feature vectors one time point by one time point.

18. The apparatus according to claim 17, wherein the proposal evaluation subunit further comprises:

the second feature buffer, used to accumulatively receive a predetermined number of basic feature vectors generated one time point by one time point and store the basic feature vectors as a basic feature sequence $S_F \in \mathbb{R}^{C \times D}$;

a boundary matching layer, used to perform point multiplication on the basic feature sequence and a sampling weight matrix $W \in \mathbb{R}^{N \times D}$, and obtain and output a boundary matching matrix $M \in \mathbb{R}^{C \times N}$ having a sampling duration N and a time region length from 1 to D , and using the time point t as an ending point; and

a convolutional neural network, the convolutional neural network comprising at least one layer, used to input the boundary matching matrix to perform convolution operation, and output the classification confidence vector $M_{cc} \in \mathbb{R}^D$ and the regression confidence vector $M_{cr} \in \mathbb{R}^D$;

wherein C denotes the number of channels of the basic feature vector, D denotes the length of the longest candidate action proposal in the basic feature sequence, the sampling weight matrix denotes the weight of the basic feature sequence S_F in the sampling duration N , M_{cc} denotes the probability of whether the candidate action proposal having a length from 1 to D and using a current time as an ending point is an action proposal, and M_{cr} denotes the probability that the candidate action proposal having a length from 1 to D and using a current time as an ending point is close to an ground truth action proposal.

19. An online temporal action detection device, comprising:

a memory, configured to store an instruction executed by one or more processors of a system, and

a processor, being one of the processors of the system, configured to execute the instruction to implement the temporal action detection method according to any one of claims 1 to 11.

20. A computer readable storage medium encoded with a computer program, wherein an instruction is stored on the computer readable medium, and when being executed on a computer, the instruction enables the computer to perform the temporal action detection method according to any one of claims 1 to 11.