

OBJECT SEQUENCE IMAGE PROCESSING METHOD AND APPARATUS, DEVICE AND STORAGE MEDIUM

TECHNICAL FIELD

[0001] The application relates to the technical field of computer vision, and relates, but not limited, to an object sequence image processing method and apparatus, a device, and a storage medium.

BACKGROUND

[0002] Sequence recognition on an image is an important research subject in computer vision. A sequence recognition algorithm is widely applied to scene text recognition, license plate recognition and other scenes. For object sequence recognition problems, the number of objects in an object sequence is uncertain, and types of the objects are diversified, so it is difficult to achieve relatively high accuracy.

[0003] A relatively effective object sequence recognition solution is to train an object sequence recognition model using labeled sample images. Since a similarity between objects forming a sequence is relatively high, and the object sequence recognition model is required to predict boundaries of single objects accurately to recognize the number and types of the objects in the sequence accurately, there is made a relatively high requirement on the labeling of the sample images.

SUMMARY

[0004] Embodiments of the application provide an object sequence image processing method and apparatus, a device, and a storage medium.

[0005] The technical solutions of the embodiments of the application are implemented

as follows.

- [0006] According to a first aspect, an embodiment of the application provides an object sequence image processing method, which may include the following operations.
- [0007] An object sequence image including at least two objects arranged in sequence is acquired.
- [0008] Edge detection is performed on the object sequence image using a trained edge detection model to obtain an edge segmentation image corresponding to the object sequence image, the edge segmentation image being configured to represent position information of an edge of each object in the object sequence image.
- [0009] Class labeling information corresponding to the object sequence image is acquired, the class labeling information including class information of each object in the object sequence image.
- [0010] Label information of the object sequence image is generated based on the edge segmentation image and the class labeling information, the label information including class information of pixels in a position region where each object is located in the object sequence image.
- [0011] According to a second aspect, an embodiment of the application provides an object sequence image processing apparatus, which may include a first acquisition module, an edge detection module, a second acquisition module, and a label generation module.
- [0012] The first acquisition module may be configured to acquire an object sequence image including at least two objects arranged in sequence.
- [0013] The edge detection module may be configured to perform edge detection on the object sequence image using a trained edge detection model to obtain an edge segmentation image corresponding to the object sequence image, the edge segmentation image being configured to represent position information of an edge of each object in the object sequence image.

- [0014] The second acquisition module may be configured to acquire class labeling information corresponding to the object sequence image, the class labeling information including class information of each object in the object sequence image.
- [0015] The label generation module may be configured to generate label information of the object sequence image based on the edge segmentation image and the class labeling information, the label information including class information of pixels in a position region where each object is located in the object sequence image.
- [0016] According to a third aspect, an embodiment of the application provides an electronic device, which may include a memory and a processor. The memory may store a computer program capable of running in the processor. The processor may execute the program to implement the steps in the object sequence image processing method described above.
- [0017] According to a fourth aspect, an embodiment of the application provides a computer-readable storage medium having stored therein a computer program which may be executed by a processor to implement the steps in the object sequence image processing method described above.
- [0018] The technical solutions provided in the embodiments of the application at least have the following beneficial effects.
- [0019] In the embodiments of the application, the object sequence image is acquired at first. Then, edge detection is performed on the object sequence image using the trained edge detection model to obtain the edge segmentation image corresponding to the object sequence image. Next, the class labeling information corresponding to the object sequence image is acquired. Finally, the label information of the object sequence image is generated based on the edge segmentation image and the class labeling information. In this manner, edge detection is performed on the object sequence image using a deep-learning-based edge detection model to obtain the edge segmentation image representing position distribution information of the objects in the object sequence image, and the pixel-level label information of the object sequence image may be automatically generated in combination with image-level object class labeling to obtain semantic labels

of the pixels in the object sequence image. Therefore, the labeling cost for semantic segmentation is reduced.

BRIEF DESCRIPTION OF THE DRAWINGS

[0020] In order to describe the technical solutions of the embodiments of the application more clearly, the drawings required to be used in the descriptions about the embodiments will be simply introduced below. It is apparent that the drawings described below are merely some embodiments of the application. Other drawings may further be obtained by those of ordinary skill in the art according to these drawings without creative work.

[0021] FIG. 1A is a flowchart of an object sequence image processing method according to an embodiment of the application.

[0022] FIG. 1B is a flowchart of an edge detection process according to an embodiment of the application.

[0023] FIG. 2A is a flowchart of an object sequence image processing method according to an embodiment of the application.

[0024] FIG. 2B is a flowchart of an object sequence image processing method according to an embodiment of the application.

[0025] FIG. 3A is a flowchart of an object sequence image processing method according to an embodiment of the application.

[0026] FIG. 3B is a flowchart of an object sequence image processing method according to an embodiment of the application.

[0027] FIG. 4 is a flowchart of an object sequence image processing method according to an embodiment of the application.

[0028] FIG. 5 is a flowchart of an object sequence image processing method according to an embodiment of the application.

[0029] FIG. 6 is a schematic diagram of comparison between an edge segmentation image output by a model and an optimized edge segmentation image according to an embodiment of the application.

[0030] FIG. 7 is a schematic diagram of an object sequence image and a corresponding semantic segmentation pseudo-label according to an embodiment of the application.

[0031] FIG. 8 is a composition structure diagram of an object sequence image processing apparatus according to an embodiment of the application.

[0032] FIG. 9 is a schematic diagram of a hardware entity of an electronic device according to an embodiment of the application.

DETAILED DESCRIPTION

[0033] In order to make the purpose, technical solutions, and advantages of the embodiments of the application clearer, the technical solutions in the embodiments of the application will be clearly and completely described below in combination with the drawings in the embodiments of the application. It is apparent that the described embodiments are not all embodiments but part of embodiments of the application. The following embodiments are adopted to describe the application rather than limit the scope of the application. All other embodiments obtained by those of ordinary skill in the art based on the embodiments in the application without creative work shall fall within the scope of protection of the application.

[0034] "Some embodiments" involved in the following descriptions describes a subset of all possible embodiments. However, it can be understood that "some embodiments" may be the same subset or different subsets of all the possible embodiments, and may be combined without conflicts.

[0035] It is to be pointed out that term "first/second/third" involved in the embodiments of the application is only for distinguishing similar objects and does not

represent a specific sequence of the objects. It can be understood that "first/second/third" may be interchanged to specific sequences or orders if allowed to implement the embodiments of the application described herein in sequences except the illustrated or described ones.

[0036] Those skilled in the art can understand that, unless otherwise defined, all the terms (including technical terms and scientific terms) used herein have the same meanings as commonly understood by those of ordinary skill in the art the embodiments of the application belong to. It should also be understood that terms defined in, for example, a general dictionary, should be understood to have the same meanings as those in the context of the conventional art, and may not be explained as idealized or too formal meanings, unless otherwise specifically defined like those here.

[0037] Computer vision, as a science researching how to make machines "see", refers to recognizing, tracking and measuring targets using video cameras and computers instead of human eyes and further performing image processing.

[0038] The basic idea of morphology is measuring or extracting a corresponding shape or feature in an input image using a special structured element (also called a convolution kernel) for further image analysis and target recognition. A morphological method is based on the set theory. An image component configured to express and describe a region shape, e.g., a boundary and a skeleton, is extracted from an image taking mathematical morphology as a tool.

[0039] Basic morphological operations include an erosion operation, a dilation operation, an opening operation (dilation after erosion), and a closing operation (erosion after dilation). Various practical mathematical morphological algorithms may further be obtained by inference and combination based on these basic operations to analyze and process image shapes and structures, including image segmentation, feature extraction, edge detection, image filtering, image enhancement and restoration, etc. A processing object of morphological transformation is usually a binary image (also a color image in many cases). A transformation process involves two inputs, i.e., an original image and a convolution kernel, and one output, i.e., a morphologically transformed image.

[0040] For the sequence recognition on an image, the following two kinds of common algorithms are adopted.

[0041] 1) A feature of the image is extracted using a Convolutional Neural Network (CNN), then sequence modeling is performed on the feature using a Recurrent Neural Network (RNN), and meanwhile, prediction and duplication elimination are performed on each feature slice under the supervision of a Connectionist Temporal Classification (CTC) loss function to obtain an output. The method has the main problems that the training of an RNN sequence modeling part is time-consuming, a model may be independently supervised using the CTC loss function only, and a prediction effect is limited.

[0042] 2) A feature of the image is extracted using a CNN, then an attention center is generated in combination with a visual attention mechanism, a corresponding result is predicted for each attention center, and other redundant information is ignored. The method has the main problem that the attention mechanism has relatively high requirements on calculations and memory usage.

[0043] An embodiment of the application provides an object sequence image processing method, which is applied to an electronic device. The electronic device includes, but not limited to, a mobile phone, a notebook computer, a pad, a palm Internet device, a multimedia device, a streaming media device, a mobile Internet device, a wearable device, or an electronic device of another type. A function realized by the method may be realized by a processor in the electronic device by calling a program code. Of course, the program code may be stored in a computer storage medium. It can be seen that the electronic device at least includes the processor and the storage medium. The processor may be configured for processing in an access control process. The memory may be configured to store data needed and generated in the access control process.

[0044] FIG. 1A is a flowchart of an object sequence image processing method according to an embodiment of the application. As shown in FIG. 1A, the method at least includes the following operations.

[0045] In S110, an object sequence image is acquired.

[0046] Here, the object sequence image includes at least two objects arranged in sequence.

[0047] In some implementation modes, the objects arranged in sequence may be photographed using a camera component arranged in different directions of the objects to obtain a real-time video, and the shot video is sent to the electronic device and sampled by the electronic device. For example, each frame of image in a sliding window may be sequentially acquired using a certain calling function, e.g., a function in a computer vision library OpenCV. In some other implementation modes, the object sequence image may also be acquired from an image or video frame collected by a local camera of the electronic device. No limits are made thereto in the embodiment of the application.

[0048] In some scenes, multiple objects forming an object sequence in the object sequence image may be separated from one another. In some other scenes, adjacent objects in multiple objects forming an object sequence in the object sequence image may be in contact with each other. For the recognition of the object sequence image, it is necessary to segment boundaries of single objects accurately and determine a class of each object in the sequence.

[0049] In S120, edge detection is performed on the object sequence image using a trained edge detection model to obtain an edge segmentation image corresponding to the object sequence image.

[0050] Here, the edge segmentation image (edge mask) is configured to represent position information of an edge of each object in the object sequence image. For example, the object is a token and the object sequence is a piled token sequence. In such case, the edge segmentation image includes a segmentation line (i.e., an edge segmentation line) after the token sequence is sliced.

[0051] The position information of each object in the object sequence image may include starting and ending positions of each object on a y axis in the object sequence image, and may further include a relative distance between adjacent objects and an arrangement order of different objects.

[0052] Optionally, the edge segmentation image includes a mask image representing

edge information of each object, and/or, the edge segmentation image is the same as the object sequence image in size. Therefore, the edge of each object in the object sequence image may be determined rapidly based on the edge segmentation image to further segment the object sequence into single objects to determine the class of each object in the object sequence.

[0053] For example, the object sequence image is a token sequence image. As shown in FIG. 1B, edge detection is performed on an original token sequence image 11 to obtain an edge segmentation image 12 on the right. The comparison between the token sequence image 11 and the edge segmentation image 12 shows that each segmentation line in the edge segmentation image 12 corresponds to an edge of a token in the token sequence image 11.

[0054] The trained edge detection model may be Richer Convolutional Feature (RCF). The model takes the object sequence image as an input and the edge segmentation image of the object as an output. The edge segmentation image is an image which is the same as the input object sequence image in size and has pixel values 0 or 1. The pixel value 1 represents that a pixel is at an edge of the object, otherwise the pixel value 0 represents that a pixel is in a region between two edge segmentation lines, i.e., a region where each object is located.

[0055] In S130, class labeling information corresponding to the object sequence image is acquired.

[0056] Here, the class labeling information includes class information of each object in the object sequence image. The class labeling information corresponding to the object sequence image is image-level object class labeling information, and indicates a class of each object in the at least two objects in the object sequence image.

[0057] In some implementation modes, for the object sequence image, the class information of each object in the object sequence image is obtained in a manual labeling manner. In some other implementation modes, a pixel-level classification task is performed on each object in the object sequence image, and then a class corresponding to most pixels in a region of the corresponding object is determined as a class of the object

by statistical voting.

[0058] Exemplarily, the labeled object sequence image includes seven objects which are arranged in sequence, the first three objects in the object sequence are labeled as class 5, while the last four objects are labeled as class 6. In such case, the classes of all the objects in the labeled object sequence image may be sequentially extracted according to the arrangement order to obtain a group of class labeling information, e.g., {class 5, class 5, class 5, class 6, class 6, class 6, class 6}.

[0059] For example, the object is a token. In some implementation modes, a class of the token is related to a face value of the token. For example, a token with a face value of 50 is labeled as class 1, and a token with a face value of 100 is labeled as class 2. In some implementation modes, the class of the token is associated with a level of a player, and classes of tokens belonging to members of different levels are different. In some implementation modes, the class of the token may also be associated with a color thereof. For example, a class of a red token is 5, and a class of a yellow token is 6.

[0060] In S140, label information of the object sequence image is generated based on the edge segmentation image and the class labeling information.

[0061] Here, the label information includes class information of pixels in a position region where each object is located in the object sequence image.

[0062] Each object in the edge segmentation image is labeled according to the class labeling information, thereby obtaining a pixel-level semantic segmentation image as the label information of the object sequence image.

[0063] It is to be noted that the object sequence image labeled with the label information may be a semantic segmentation image, i.e., an image including edge segmentation information and semantic segmentation information. Therefore, the object sequence image labeled with the label information may be directly input to a deep-learning-based semantic segmentation neural network as training data to complete a pixel-level classification task for the input object sequence image to further complete the recognition of the class of each object in the object sequence.

[0064] In the embodiment of the application, the object sequence image is acquired at first. Then, edge detection is performed on the object sequence image using the trained edge detection model to obtain the edge segmentation image corresponding to the object sequence image. Next, the class labeling information corresponding to the object sequence image is acquired. Finally, the label information of the object sequence image is generated based on the edge segmentation image and the class labeling information. In this manner, edge detection is performed on the object sequence image using a deep-learning-based edge detection model to obtain the edge segmentation image representing position distribution information of the objects in the object sequence image, and the pixel-level label information of the object sequence image may be automatically generated in combination with image-level object class labeling. Therefore, the labeling cost for semantic segmentation is reduced.

[0065] FIG. 2A is a flowchart of an object sequence image processing method according to an embodiment of the application. As shown in FIG. 2A, the method at least includes the following operations.

[0066] In S201, an object sequence image is acquired.

[0067] In S202, edge detection is performed on the object sequence image using a trained edge detection model to obtain an edge segmentation image corresponding to the object sequence image.

[0068] In S203, class labeling information corresponding to the object sequence image is acquired.

[0069] In S204, a position region where each object is located is determined based on position information of an edge of each object in the edge segmentation image.

[0070] Here, the edge segmentation image is configured to represent the position information of the edge of each object in the object sequence image. The edge segmentation image includes segmentation lines after an object sequence is sliced. A region between different segmentation lines is a position region where the corresponding object is located.

[0071] In some implementation modes, logic processing may be performed on the edge segmentation image to obtain the number of the segmentation lines and a position of each segmentation line, i.e., a coordinate of the segmentation line. Meanwhile, the number of the objects in the object sequence image, a position region corresponding to each object on a y axis in the whole object sequence image and a center coordinate of the object may be determined based on the number of edge lines.

[0072] In S205, classes of non-object-edge pixels in the position region where each object is located in the edge segmentation image are labeled according to class information of each object.

[0073] Here, the non-object-edge pixel refers to a pixel in the position region where the object is located. In some implementation modes, a labeling process may be implemented through the following operations. A coded value corresponding to an object of each class is determined based on the class labeling information. Corresponding pixels in the position region where the corresponding object is located between adjacent segmentation lines are labeled as the coded value. Therefore, a labeled object sequence image including pixel-level semantic segmentation information may be obtained rapidly based on image-level object class labeling information for large-scale model training.

[0074] It can be understood that, if a digital image includes $m*n$ pixels, the image is segmented into $m*n$ squares by row lines numbered as 0 to m and column lines numbered as 0 to n . Each square is called a pixel, and a region of the whole square has the same gray-scale value, i.e., pixel value of the square. Therefore, the pixel values in the position region where the corresponding object is located between adjacent segmentation lines may be labeled as a coded value of a class of the object.

[0075] In S206, label information of the object sequence image is generated according to a labeling result of the classes of the non-object-edge pixels.

[0076] Here, the classes of the non-object-edge pixels in the edge segmentation image are labeled according to the class labeling information to obtain a pixel-level semantic segmentation image, so that the label information of the object sequence image may further be generated.

[0077] In some implementation modes, the edge segmentation image includes a mask image representing edge information of each object, and/or, the edge segmentation image is the same as the object sequence image in size. In this manner, the object between every two segmentation lines in the edge segmentation image may be directly labeled to generate the label information of the object sequence image, thereby obtaining an image including both edge segmentation information and semantic segmentation information.

[0078] In the embodiment of the application, the pixels in each object in the edge segmentation image are labeled using the class labeling information corresponding to the acquired object sequence image, thereby automatically generating pixel-level label information. Label information of a large number of object sequence images may be obtained as training data to improve the recognition accuracy of a semantic-segmentation-based object recognition method under the condition of avoiding manual pixel-level classification labeling and high labor cost and time cost.

[0079] FIG. 2B is a flowchart of an object sequence image processing method according to an embodiment of the application. As shown in FIG. 2B, the method at least includes the following operations.

[0080] In S211, an object sequence image is acquired.

[0081] In S212, edge detection is performed on the object sequence image using a trained edge detection model to obtain an edge segmentation image corresponding to the object sequence image.

[0082] In S213, class labeling information corresponding to the object sequence image is acquired.

[0083] In S214, a position region where each object is located is determined based on position information of an edge of each object in the edge segmentation image.

[0084] In S215, classes of non-object-edge pixels in the position region where each object is located in the edge segmentation image are labeled according to class information of each object.

- [0085] In S216, classes of edge pixels in the edge segmentation image are labeled as a preset class based on the position information of the edge of each object in the edge segmentation image.
- [0086] Here, the preset class is different from the class of each non-object-edge pixel.
- [0087] In some implementation modes, a pixel value at a corresponding position of a segmentation line is labeled as a coded value represented by the segmentation line, and a pixel value representing the position region of an object between adjacent segmentation lines is labeled as a coded value corresponding to a class of the corresponding object.
- [0088] In S217, label information of the object sequence image is generated according to a labeling result of the classes of the non-object-edge pixels and a labeling result of the classes of the object edge pixels.
- [0089] Here, the classes of the non-object-edge pixels in the edge segmentation image are labeled according to the class labeling information, and meanwhile, the classes of the object edge pixels are labeled according to the preset class, to obtain a pixel-level semantic segmentation image, so that the label information of the object sequence image may further be completed. That is, the class information of each object and edge information of each object in the edge segmentation image may be obtained through the same semantic segmentation image.
- [0090] In the embodiment of the application, the pixels in each object and the edge of each object in the edge segmentation image are labeled using the class labeling information corresponding to the acquired object sequence image and the preset class respectively, thereby automatically generating pixel-level label information. Label information of a large number of object sequence images may be obtained as training data to improve the recognition accuracy of a semantic-segmentation-based object recognition method under the condition of avoiding manual pixel-level classification labeling and high labor cost and time cost.
- [0091] FIG. 3A is a flowchart of an object sequence image processing method according to an embodiment of the application. As shown in FIG. 3A, the method may be implemented through the following operations.

- [0092] In S301, an object sequence image is acquired.
- [0093] In S302, edge detection is performed on the object sequence image using a trained edge detection model to obtain an edge segmentation image corresponding to the object sequence image.
- [0094] In S303, class labeling information corresponding to the object sequence image is acquired.
- [0095] Here, the class labeling information includes class information of each object in the object sequence image.
- [0096] In S304, morphological transformation and edge segmentation line thickness uniformization are performed on the edge segmentation image to obtain an optimized edge segmentation image.
- [0097] Here, the morphological transformation is used to remove redundant noises in the edge segmentation image and connect broken parts of segmentation lines in the edge segmentation image.
- [0098] The edge segmentation line thickness uniformization may be implemented by skeletonization. The skeletonization is used to limit the thickness of each segmentation line in the edge segmentation image, for example, reducing the thickness of each segmentation line, and adjust the thicknesses of all the segmentation lines to the same scale for subsequent processing.
- [0099] The thickness of each segmentation line in the optimized edge segmentation image obtained by sequentially performing morphological transformation and edge segmentation line thickness uniformization on the edge segmentation image output by the edge detection model is controlled in the same scale to facilitate the subsequent accurate semantic labeling of the optimized edge segmentation image.
- [00100] In S305, label information of the object sequence image is generated according to the class labeling information and the optimized edge segmentation image.
- [00101] Here, a pixel value of an object corresponding to each position in the

optimized edge segmentation image is labeled as a class of the corresponding object according to the class labeling information to generate a pixel-level semantic segmentation image, so that the semantic segmentation image combining edge information is determined as the label information of the object sequence image.

[00102] In the embodiment of the application, morphological transformation and edge segmentation line thickness uniformization are performed on the edge segmentation image output by the edge detection model to obtain the optimized edge segmentation image, and then the optimized edge segmentation image is labeled according to the class labeling information. As such, the label information of the object sequence image may be automatically generated. The label information, generated in the embodiment of the application, of the object sequence image includes both edge segmentation information and semantic segmentation information, and may be directly used as training data to complete a pixel-level classification task for the object sequence image to further recognize the class of each object in the object sequence.

[00103] FIG. 3B is a flowchart of an object sequence image processing method according to an embodiment of the application. As shown in FIG. 3B, the method may be implemented through the following operations.

[00104] In S311, an object sequence image is acquired.

[00105] In S312, an edge detection model is trained based on a first object sequence sample image.

[00106] Here, the first object sequence sample image includes edge segmentation information of each object forming the first object sequence sample image.

[00107] In some implementation modes, edge label information of the first object sequence sample image in an edge detection module is determined using an optimized edge segmentation image, so that training data for the edge detection model may be continuously extended.

[00108] In some implementation modes, the first object sequence sample image and the edge label information are input together to a fully CNN for training, so that an effect of

the edge segmentation image output by a final edge detection model is continuously improved, and an effect of label information, generated based on the edge segmentation image, of the object sequence image sample is also continuously improved.

[00109] In S313, edge detection is performed on the object sequence image using a trained edge detection model to obtain an edge segmentation image corresponding to the object sequence image.

[00110] In S314, class labeling information corresponding to the object sequence image is acquired.

[00111] Here, the class labeling information includes class information of each object in the object sequence image.

[00112] In S315, label information of the object sequence image is generated according to the class labeling information and the edge segmentation image.

[00113] In S316, an object sequence recognition model is trained taking the object sequence image including the label information as a second object sequence sample image.

[00114] Here, the object sequence recognition model is configured to recognize an object sequence based on an image. The object sequence image including the label information, obtained by the method of the abovementioned embodiment, is input to a CNN as the second object sequence sample image, and training is performed to obtain a trained object sequence recognition model, so that pixel-level classification prediction may be performed on an object in an object sequence image to be detected to implement object sequence recognition, and a relatively accurate recognition result may be obtained.

[00115] In the embodiment of the application, the first object sequence sample image is optimized, and the edge label information is determined, so that the effect of the edge segmentation image output by the final edge detection model is continuously improved, and furthermore, the semantic segmentation effect of the label information of the object sequence image sample generated based on the edge segmentation image is improved. The label information, generated in the embodiment of the application, of the object

sequence image includes both edge segmentation information and semantic segmentation information, and may be directly used as the second object sequence sample image to train the object sequence recognition model to complete a pixel-level classification task for the object sequence image to further recognize the class of each object in the object sequence.

[00116] FIG. 4 is a flowchart of an object sequence image processing method according to an embodiment of the application. As shown in FIG. 4, the operation in S304 that "morphological transformation and edge segmentation line thickness uniformization are performed on the edge segmentation image to obtain an optimized edge segmentation image" may be implemented through the following operations.

[00117] In S410, denoising processing is performed on the edge segmentation image using a first morphological operation to obtain a denoised edge segmentation image.

[00118] It is to be noted that the edge segmentation image output by the edge detection model usually has a relatively poor effect and the problems of breaking points, non-uniform thickness, non-ignorable noises, etc., and it is necessary to further perform processes of denoising, connecting broken parts of segmentation lines, adjusting thicknesses of segmentation lines, etc., using morphological operations.

[00119] Here, the first morphological operation is an opening operation (morphologyEx opening) in OpenCV. The edge segmentation image output by the edge detection model and a convolution kernel are taken as an input of the opening operation, and the denoised edge segmentation image may be obtained.

[00120] Erosion and dilation processing are sequentially performed in the opening operation. Dilation refers to calculating a local maximum. The edge segmentation image and the convolution kernel are convolved, and a specified pixel is assigned with the maximum, thereby making a bright region brighter with an effect of dilating the bright region. Opposite to a dilation operation, an erosion operation refers to calculating a local minimum.

[00121] It is to be noted that the opening operation refers to sequentially performing erosion and dilation. Generally, erosion and dilation are sequentially performed during

denoising. This is because erosion may reduce a foreground target at the same time of removing white noises. When an eroded edge segmentation image is dilated, the noises have been removed and no longer exist, but the foreground still exists and may be enlarged. Since a noise has a contour, the noise may be removed at first using an erosion operation, and then the edge segmentation image is "restored" by a dilation operation. In this manner, redundant noises in the edge segmentation image output by the edge detection model may be removed effectively.

[00122] In S420, dilation processing is performed on the denoised edge segmentation image using a second morphological operation to obtain an edge segmentation image with continuous segmentation line.

[00123] Here, the second morphological operation corresponds to a dilation function in OpenCV. The denoised edge segmentation image and a convolution kernel are taken as an input of the dilation function, and an output pixel value is a maximum pixel value of an input image under the coverage of the convolution kernel, i.e., a structured element, (255 is white in a gray-scale image).

[00124] It is to be noted that a pixel value of a central element is 1 if one of pixel values in the original denoised edge segmentation image corresponding to the convolution kernel is 1, so the dilation operation may enlarge a white region (foreground) in the image. Dilation may also be used to connect two separate objects. When the convolution kernel moves to the vicinity of a contour of a segmentation line during the dilation operation, a value of a center pixel is set to 1 if one of the pixel values in the convolution kernel is 1. When the convolution kernel is at an edge of the image, value 1 may also exist in the convolution kernel, and the center of the convolution kernel is set to 1, so that the image is extended by a circle with a visual effect of dilating the image.

[00125] In S430, erosion processing is performed on the edge segmentation image with continuous segmentation line using a third morphological operation to obtain a transformed edge segmentation image.

[00126] Here, the third morphological operation corresponds to an erosion function in OpenCV. The edge segmentation image with continuous segmentation line and a

convolution kernel are taken as an input of the erosion function, and the transformed edge segmentation image may be output by processing.

[00127] The convolution kernel may slide along the edge segmentation image with continuous segmentation line. If all pixel values of the edge segmentation image with continuous segmentation line corresponding to the convolution kernel are 1, a center pixel keeps an original pixel value, otherwise changes to zero. When the convolution kernel moves to the vicinity of a contour during the erosion operation, the original pixel value is kept if all values in the convolution kernel are the same. When there are different pixel values (a half of the convolution kernel is in the contour, while the other half is in a background image), the erosion operation converts the different pixel values into the same value as the contour, so that an area of a black contour is enlarged with a visual effect of erosion.

[00128] In some implementation modes, the convolution kernel for the first morphological operation is set to 5×5 to ensure the effective removal of the noises in the edge segmentation image by the first morphological operation. The convolution kernel for the second morphological operation is set to 3×10 to ensure the connection of horizontally broken parts of connecting lines in the denoised edge segmentation image and the thickening of the segmentation lines by the second morphological operation. The convolution kernel for the third morphological operation is set to 3×3 to reduce the thickness of the segmentation line to an original size.

[00129] In the embodiment of the application, the morphological operations in OpenCV are used to remove the redundant noises in the edge segmentation image at first and then connect the broken parts of the segmentation lines in the edge segmentation image to obtain the transformed edge segmentation image.

[00130] In S440, skeletonization processing is performed on the transformed edge segmentation image to uniformize the thickness of a segmentation line to obtain the optimized edge segmentation image.

[00131] Here, the thickness of each segmentation line in the optimized edge segmentation image is controlled in the same scale.

- [00132] Image skeletonization, also called thinning, usually refers to an operation for skeletonizing a binary image. A purpose of the skeletonization processing is to further reduce the thickness of the segmentation line and simultaneously adjust the thickness of each segmentation line to the same scale to facilitate the subsequent accurate semantic labeling of the optimized edge segmentation image.
- [00133] In the embodiment of the application, the edge segmentation image output by the edge detection model is processed sequentially using the opening operation, dilation function and erosion function in OpenCV to finally obtain a redundant noise-free edge segmentation image of which the thicknesses of the segmentation lines are uniform, continuous and in the same scale. Therefore, the semantic segmentation effect of the label information, generated based on the edge segmentation image, of the object sequence image is improved.
- [00134] The object sequence image processing method will be described below in combination with a specific embodiment. However, it is to be noted that the specific embodiment is only for describing the application better and does not form improper limits to the application.
- [00135] Descriptions are made in the embodiment of the application with an example that the object sequence image is a token sequence image in an entertainment place. FIG. 5 is a flowchart of an object sequence image processing method according to an embodiment of the application. As shown in FIG. 5, the method includes the following operations.
- [00136] In S510, an original edge segmentation image is generated using a trained edge detection model.
- [00137] During implementation, an edge detection model is pre-trained using a small number of labeled edge segmentation images (about 2,000) to generate an edge segmentation image.
- [00138] The token sequence image is input to the trained edge detection model, and an edge segmentation image corresponding to the token sequence image is output. The token sequence image and the original edge segmentation image are shown in FIG. 1B

respectively. It can be seen that each segmentation line in the edge segmentation image 12 corresponds to an edge of a token in the token sequence image 11.

[00139] In S520, morphological transformation is performed on the original edge segmentation image.

[00140] As shown in the left figure in FIG. 6, an edge segmentation image 61 output by the edge detection model has a relatively poor effect and the problems of breaking points, non-uniform thickness, non-ignorable noises, etc. For the edge segmentation image obtained in S510, two processes of removing redundant noises and connecting broken segmentation lines are performed respectively in the embodiment of the application.

[00141] In the process of removing the redundant noises, an opening operation is adopted, and a convolution kernel is set to 5×5 in this function. In the process of connecting the broken segmentation lines, a dilation function in OpenCV is used at first, a convolution kernel therein being set to 3×10 . This function may connect horizontally connected parts of the image and simultaneously thicken the segmentation lines. Then, an erosion function in OpenCV is used, a convolution kernel therein being set to 3×3 . The thickness of the segmentation line may be reduced to an original size.

[00142] In S530, skeletonization processing is performed on a transformed edge segmentation image.

[00143] Here, skeletonization refers to stripping layer by layer. Some pixels are removed from the original image, but an original shape is required to be kept, until a skeleton of the image is obtained. A purpose of the skeletonization processing is to further reduce the thickness of the segmentation line and simultaneously adjust the thickness of each segmentation line to the same scale to facilitate subsequent processing. An optimized edge segmentation image is shown in the right figure in FIG. 6. The optimized edge segmentation image 62 is noiseless, the segmentation lines are continuous, and the thickness of each segmentation line is controlled in the same scale.

[00144] The skeletonization processing is implemented through a skeletonization flow in OpenCV. The flow includes the following operations.

- [00145] In S1, a blank skeleton image the same as the input edge segmentation image in size is created, all pixel values in the image being 0. A convolution kernel for an operation is defined as a 3*3 cross convolution kernel.
- [00146] In S2, the transformed edge segmentation image is processed using an opening operation in OpenCV to obtain a second edge segmentation image.
- [00147] In S3, subtraction is performed on the transformed edge segmentation image and the second edge segmentation image to obtain pixels removed in the opening operation as a candidate image.
- [00148] In S4, the candidate image and a present blank skeleton image are added to obtain a new skeleton image.
- [00149] In S5, an erosion operation is performed on the transformed edge segmentation image, and an erosion result is redefined as a new edge segmentation image to prepare for further skeleton extraction.
- [00150] In S6, S2 to S5 are repeated until all pixel values in a new edge segmentation image are 0, namely the new edge segmentation image is completely eroded. A present skeleton image is an extracted final skeleton image, equivalent to the optimized edge segmentation image.
- [00151] It is to be noted that the optimized edge segmentation image may be taken as a pseudo-label of an edge segmentation model to continuously extend training data for the edge detection model to continuously improve the effect of the segmentation image output by the final edge detection model and the effect of the semantic segmentation pseudo-label.
- [00152] In S540, a semantic segmentation pseudo-label is generated using a processed edge segmentation image and a token class labeling image.
- [00153] Here, the semantic segmentation pseudo-label is equivalent to label information of the token sequence image, and may be used for large-scale token recognition model training.

[00154] The token class labeling image indicates image-level token class labeling information. For example, {class 5, class 5, class 5, class 6, class 6, class 6, class 6} indicates that first three tokens in a token sequence are tokens of class 5 while last four tokens are tokens of class 6. As shown in FIG. 7, the right figure is the token sequence image, and the left figure is the generated semantic segmentation pseudo-label. A pixel value at a corresponding position of a segmentation line 70 is labeled as a coded value represented by the segmentation line. A pixel value representing a position of a token between the segmentation lines 70 is labeled as a coded value corresponding to a class of the token. For example, different colors represent that a pixel value of a token 71, a pixel value of a token 72 and a pixel value of a token 73, which belong to different classes, are labeled as coded values of the corresponding classes respectively.

[00155] According to the embodiment of the application, a large number of pixel-level semantic segmentation pseudo-labels are automatically generated for large-scale token recognition model training by training the edge detection model and performing sequence image processing optimization based on a small amount of edge segmentation labeling and image-level token class labeling.

[00156] According to the embodiment of the application, pixel-level pseudo-labels are generated based on the image-level labeling information to obtain a large amount of training data under the condition of avoiding high labor cost and time cost and further improve the recognition accuracy of a semantic-segmentation-based token recognition method.

[00157] In an intelligent game place, the embodiment of the application may be combined with the semantic-segmentation-based token recognition method to count and recognize tokens in a token sequence image better, thereby ensuring that a payout flow, counting of the tokens and the like are implemented more conveniently and rapidly and saving manpower.

[00158] The object sequence image processing method provided in the embodiment of the application may be applied to a casino scene. In the casino scene, the object sequence image mentioned anywhere in the embodiment of the application may be a chip sequence image obtained by shooting chips on a gaming table in the casino, the player mentioned

anywhere in the embodiment of the application may include the player or the banker, the game place mentioned anywhere in the embodiment of the application may refer to the casino, and the token mentioned anywhere in the embodiment of the application may include chip.

[00159] It is to be noted that each embodiment in the specification is described progressively. Descriptions made in each embodiment focus on differences with the other embodiments and the same or similar parts in each embodiment refer to the other embodiments.

[00160] Based on the abovementioned embodiment, an embodiment of the application also provides an object sequence image processing apparatus. Each module of the apparatus and each submodule and each unit of each module may be implemented by a processor in an electronic device, and of course, may also be implemented by a specific logic circuit. In an implementation process, the processor may be a Central Processing Unit (CPU), a Micro Processing Unit (MPU), a Digital Signal Processor (DSP), a Field Programmable Gate Array (FPGA), or the like.

[00161] FIG. 8 is a composition structure diagram of an object sequence image processing apparatus according to an embodiment of the application. As shown in FIG. 8, the apparatus 800 includes a first acquisition module 810, an edge detection module 820, a second acquisition module 830, and a label generation module 840.

[00162] The first acquisition module 810 is configured to acquire an object sequence image including at least two objects arranged in sequence.

[00163] The edge detection module 820 is configured to perform edge detection on the object sequence image using a trained edge detection model to obtain an edge segmentation image corresponding to the object sequence image, the edge segmentation image being configured to represent position information of an edge of each object in the object sequence image.

[00164] The second acquisition module 830 is configured to acquire class labeling information corresponding to the object sequence image, the class labeling information including class information of each object in the object sequence image.

[00165] The label generation module 840 is configured to generate label information of the object sequence image based on the edge segmentation image and the class labeling information, the label information including class information of pixels in a position region where each object is located in the object sequence image.

[00166] In some possible embodiments, the label generation module 840 includes: a first determination submodule, configured to determine the position region where each object is located based on the position information of the edge of each object in the edge segmentation image; a first labeling submodule, configured to label classes of non-object-edge pixels in the position region where each object is located in the edge segmentation image according to the class information of each object; and a first generation submodule, configured to generate the label information of the object sequence image according to a labeling result of the classes of the non-object-edge pixels.

[00167] In some possible embodiments, the label generation module 840 further includes a second labeling submodule, configured to label classes of object edge pixels in the edge segmentation image as a preset class based on the position information of the edge of each object in the edge segmentation image, the preset class being different from the class of each non-object-edge pixel. The first generation submodule is further configured to generate the label information of the object sequence image according to the labeling result of the classes of the non-object-edge pixels and a labeling result of the classes of the object edge pixels.

[00168] In some possible embodiments, the first generation submodule is further configured to generate a semantic segmentation image the same as the object sequence image in size according to the labeling result of the class of each non-object-edge pixel and the labeling result of each object edge pixel in the object sequence image, a pixel value of each pixel in the semantic segmentation image being configured to identify a class of each pixel.

[00169] In some possible embodiments, the label generation module 840 includes: an optimization submodule, configured to perform morphological transformation and edge segmentation line thickness uniformization on the edge segmentation image to obtain an optimized edge segmentation image; and a second generation submodule, configured to

generate the label information of the object sequence image according to the class labeling information and the optimized edge segmentation image.

[00170] In some possible embodiments, the optimization submodule includes: a first processing subunit, configured to perform denoising processing on the edge segmentation image using a first morphological operation to obtain a denoised edge segmentation image; a second processing subunit, configured to perform dilation processing on the denoised edge segmentation image using a second morphological operation to obtain an edge segmentation image with continuous segmentation line; a third processing subunit, configured to perform erosion processing on the edge segmentation image with continuous segmentation line using a third morphological operation to obtain a transformed edge segmentation image; and a skeletonization subunit, configured to perform skeletonization processing on the transformed edge segmentation image to uniformize the thickness of a segmentation line to obtain the optimized edge segmentation image, the thickness of each segmentation line in the optimized edge segmentation image being controlled in the same scale.

[00171] In some possible embodiments, the apparatus 800 further includes a first model training module, configured to train an edge detection model based on a first object sequence sample image including edge segmentation information of each object forming a first object sequence.

[00172] In some possible embodiments, the apparatus 800 further includes a second model training module, configured to train an object sequence recognition model taking the object sequence image including the label information as a second object sequence sample image, the object sequence recognition model being configured to recognize an object sequence based on an image.

[00173] It is to be pointed out that descriptions about the above apparatus embodiment are similar to those about the method embodiment, and beneficial effects similar to those of the method embodiment are achieved. Technical details undisclosed in the apparatus embodiment of the application may be understood with reference to the descriptions about the method embodiment of the application.

[00174] It is to be noted that, in the embodiments of the application, the object sequence image processing method may also be stored in a computer-readable storage medium when implemented in form of software function module and sold or used as an independent product. Based on such an understanding, the technical solutions of the embodiments of the application substantially or parts making contributions to the related art may be embodied in form of software product. The computer software product is stored in a storage medium, including a plurality of instructions configured to enable an electronic device (which may be a smart phone with a camera, a tablet computer, etc.) to execute all or part of the method in each embodiment of the application. The storage medium includes various media capable of storing program codes such as a U disk, a mobile hard disk, a Read Only Memory (ROM), a magnetic disk, or an optical disk. Therefore, the embodiments of the application are not limited to any specific hardware and software combination.

[00175] Correspondingly, an embodiment of the application provides a computer-readable storage medium having stored therein a computer program which is executed by a processor to implement the steps in the object sequence image processing method in any abovementioned embodiment. Correspondingly, an embodiment of the application also provides a chip, which includes a programmable logic circuit and/or a program instruction and is configured to, when running, implement the steps in the object sequence image processing method in any abovementioned embodiment. Correspondingly, an embodiment of the application also provides a computer program product which, when executed by a processor of an electronic device, is configured to implement the steps in the object sequence image processing method in any abovementioned embodiment.

[00176] Based on the same technical concept, an embodiment of the application provides an electronic device, which is configured to implement the object sequence image processing method recorded in the method embodiment. FIG. 9 is a schematic diagram of a hardware entity of an electronic device according to an embodiment of the application. As shown in FIG. 9, the electronic device 900 includes a memory 910 and a processor 920. The memory 910 stores a computer program capable of running in the processor 920. The processor 920 executes the program to implement the steps in any

object sequence image processing method as described in the embodiments of the application.

[00177] The memory 910 is configured to store an instruction and application executable for the processor 920, may also cache data (for example, image data, video data, voice communication data, and video communication data) to be processed or having been processed by the processor 920 and each module in the electronic device 400, and may be implemented through a flash or a Random Access Memory (RAM).

[00178] The processor 920 executes the program to implement the steps of any abovementioned object sequence image processing method. The processor 920 usually controls overall operations of the electronic device 900.

[00179] The processor may be at least one of an Application Specific Integrated Circuit (ASIC), a DSP, a Digital Signal Processing Device (DSPD), a Programmable Logic Device (PLD), an FPGA, a CPU, a controller, a microcontroller, or an MPU. It can be understood that other electronic devices may also be configured to realize functions of the processor, and no specific limits are made in the embodiments of the application.

[00180] The computer storage medium/memory may be a memory such as a ROM, a Programmable Read-Only Memory (PROM), an Erasable Programmable Read-Only Memory (EPROM), an Electrically Erasable Programmable Read-Only Memory (EEPROM), a Ferromagnetic Random Access Memory (FRAM), a flash memory, a magnetic surface memory, an optical disk, or a Compact Disc Read-Only Memory (CD-ROM), or may be any electronic device including one or any combination of the abovementioned memories, such as a mobile phone, a computer, a tablet device, and a personal digital assistant.

[00181] It is to be pointed out here that the above descriptions about the storage medium and device embodiments are similar to those about the method embodiment, and beneficial effects similar to those of the method embodiment are achieved. Technical details undisclosed in the storage medium and device embodiment of the application are understood with reference to the descriptions about the method embodiment of the application.

[00182] It is to be understood that "one embodiment" and "an embodiment" mentioned in the whole specification mean that specific features, structures or characteristics related to the embodiment is included in at least one embodiment of the application. Therefore, "in one embodiment" or "in an embodiment" mentioned throughout the specification does not always refer to the same embodiment. In addition, these specific features, structures or characteristics may be combined in one or more embodiments freely as appropriate. It is to be understood that, in each embodiment of the application, the magnitude of the sequence number of each process does not mean an execution sequence and the execution sequence of each process should be determined by its function and an internal logic and should not form any limit to an implementation process of the embodiments of the application. The sequence numbers of the embodiments of the application are adopted not to represent superiority-inferiority of the embodiments but only for description.

[00183] It is to be noted that terms "include" and "contain" or any other variant thereof is intended to cover nonexclusive inclusions herein, so that a process, method, object, or device including a series of elements not only includes those elements but also includes other elements which are not clearly listed or further includes elements intrinsic to the process, the method, the object, or the device. Under the condition of no more limitations, an element defined by the statement "including a/an....." does not exclude existence of the same other elements in a process, method, object, or device including the element.

[00184] In some embodiments provided by the application, it is to be understood that the disclosed device and method may be implemented in another manner. The device embodiment described above is only schematic, and for example, division of the units is only logic function division, and other division manners may be adopted during practical implementation. For example, multiple units or components may be combined or integrated into another system, or some characteristics may be neglected or not executed. In addition, coupling or direct coupling or communication connection between each displayed or discussed component may be indirect coupling or communication connection, implemented through some interfaces, of the device or the units, and may be electrical and mechanical or adopt other forms.

[00185] The units described as separate parts may or may not be physically separated, and parts displayed as units may or may not be physical units, and namely may be located in the same place, or may also be distributed to multiple network units. Part of all of the units may be selected according to a practical requirement to achieve the purposes of the solutions of the embodiments of the application.

[00186] In addition, each function unit in each embodiment of the application may be integrated into a processing unit, each unit may also serve as an independent unit and two or more than two units may also be integrated into a unit. The integrated unit may be implemented in a hardware form and may also be implemented in form of hardware and software function unit.

[00187] Or, the integrated unit of the application may also be stored in a computer-readable storage medium when implemented in form of a software function module and sold or used as an independent product. Based on such an understanding, the technical solutions of the embodiments of the application substantially or parts making contributions to the related art may be embodied in form of a software product. The computer software product is stored in a storage medium, including a plurality of instructions configured to enable an automatic test line of a device to execute all or part of the method in each embodiment of the application. The storage medium includes various media capable of storing program codes such as a mobile hard disk, a ROM, a magnetic disk, or an optical disc.

[00188] The methods disclosed in some method embodiments provided in the application may be freely combined without conflicts to obtain new method embodiments.

[00189] The characteristics disclosed in some method or device embodiments provided in the application may be freely combined without conflicts to obtain new method embodiments or device embodiments.

[00190] The above is only the implementation mode of the application and not intended to limit the scope of protection of the application. Any variations or replacements apparent to those skilled in the art within the technical scope disclosed by the application

shall fall within the scope of protection of the application. Therefore, the scope of protection of the application shall be subject to the scope of protection of the claims.

CLAIMS

1. An object sequence image processing method, comprising:

acquiring an object sequence image comprising at least two objects arranged in sequence;

performing edge detection on the object sequence image using a trained edge detection model to obtain an edge segmentation image corresponding to the object sequence image, the edge segmentation image being configured to represent position information of an edge of each object in the object sequence image;

acquiring class labeling information corresponding to the object sequence image, the class labeling information comprising class information of the each object in the object sequence image; and

generating label information of the object sequence image based on the edge segmentation image and the class labeling information, the label information comprising class information of pixels in a position region where the each object is located in the object sequence image.

2. The method of claim 1, wherein the generating label information of the object sequence image based on the edge segmentation image and the class labeling information comprises:

determining the position region where the each object is located based on the position information of the edge of the each object in the edge segmentation image;

labeling classes of non-object-edge pixels in the position region where the each object is located in the edge segmentation image according to the class information of the each object; and

generating the label information of the object sequence image according to a labeling result of the classes of the non-object-edge pixels.

3. The method of claim 2, wherein the generating label information of the object sequence image based on the edge segmentation image and the class labeling information further comprises:

labeling classes of object edge pixels in the edge segmentation image as a preset class based on the position information of the edge of the each object in the edge segmentation image, the preset class being different from the class of each non-object-edge pixel; and

wherein the generating the label information of the object sequence image according to a labeling result of the classes of the non-object-edge pixels comprises:

generating the label information of the object sequence image according to the labeling result of the classes of the non-object-edge pixels and a labeling result of the classes of the object edge pixels.

4. The method of claim 3, wherein the generating the label information of the object sequence image according to the labeling result of the classes of the non-object-edge pixels and a labeling result of the classes of the object edge pixels comprises:

generating a semantic segmentation image the same as the object sequence image in size according to the labeling result of the class of the each non-object-edge pixel and the labeling result of each object edge pixel in the object sequence image, a pixel value of each pixel in the semantic segmentation image being configured to identify a class of the each pixel.

5. The method of any one of claims 1-4, wherein the generating label information of the object sequence image based on the edge segmentation image and the class labeling information comprises:

performing morphological transformation and edge segmentation line thickness uniformization processing on the edge segmentation image to obtain an optimized edge

segmentation image; and

generating the label information of the object sequence image according to the class labeling information and the optimized edge segmentation image.

6. The method of claim 5, wherein the performing morphological transformation and edge segmentation line thickness uniformization processing on the edge segmentation image to obtain an optimized edge segmentation image comprises:

performing denoising processing on the edge segmentation image using a first morphological operation to obtain a denoised edge segmentation image;

performing dilation processing on the denoised edge segmentation image using a second morphological operation to obtain an edge segmentation image with continuous segmentation line;

performing erosion processing on the edge segmentation image with continuous segmentation line using a third morphological operation to obtain a transformed edge segmentation image; and

performing skeletonization processing on the transformed edge segmentation image to uniformize a thickness of a segmentation line to obtain the optimized edge segmentation image.

7. The method of any one of claims 1-6, wherein the edge segmentation image comprises a mask image representing edge information of the each object, and/or, the edge segmentation image is the same as the object sequence image in size.

8. The method of any one of claims 1-7, further comprising:

training an edge detection model based on a first object sequence sample image

comprising edge segmentation information of the each object forming the first object sequence sample image.

9. The method of any one of claims 1-8, further comprising:

training an object sequence recognition model by taking the object sequence image comprising the label information as a second object sequence sample image, the object sequence recognition model being configured to recognize an object sequence based on an image.

10. An electronic device, comprising a memory and a processor, wherein the memory stores a computer program capable of running in the processor, wherein when executing the compute program, the processor is configured to:

acquire an object sequence image comprising at least two objects arranged in sequence;

perform edge detection on the object sequence image using a trained edge detection model to obtain an edge segmentation image corresponding to the object sequence image, the edge segmentation image being configured to represent position information of an edge of each object in the object sequence image;

acquire class labeling information corresponding to the object sequence image, the class labeling information comprising class information of the each object in the object sequence image; and

generate label information of the object sequence image based on the edge segmentation image and the class labeling information, the label information comprising class information of pixels in a position region where the each object is located in the object sequence image.

11. The electronic device of claim 10, wherein when generating the label information of the

object sequence image based on the edge segmentation image and the class labeling information, the processor is configured to:

determine the position region where the each object is located based on the position information of the edge of the each object in the edge segmentation image;

label classes of non-object-edge pixels in the position region where the each object is located in the edge segmentation image according to the class information of the each object; and

generate the label information of the object sequence image according to a labeling result of the classes of the non-object-edge pixels.

12. The electronic device of claim 11, wherein when generating the label information of the object sequence image based on the edge segmentation image and the class labeling information, the processor is further configured to:

label classes of object edge pixels in the edge segmentation image as a preset class based on the position information of the edge of the each object in the edge segmentation image, the preset class being different from the class of each non-object-edge pixel; and

wherein when generating the label information of the object sequence image according to a labeling result of the classes of the non-object-edge pixels, the processor is configured to:

generate the label information of the object sequence image according to the labeling result of the classes of the non-object-edge pixels and a labeling result of the classes of the object edge pixels.

13. The electronic device of claim 12, wherein when generating the label information of the object sequence image according to the labeling result of the classes of the non-object-edge pixels and the labeling result of the classes of the object edge pixels, the processor is configured to:

generate a semantic segmentation image the same as the object sequence image in size according to the labeling result of the class of the each non-object-edge pixel and the labeling result of each object edge pixel in the object sequence image, a pixel value of each pixel in the semantic segmentation image being configured to identify a class of the each pixel.

14. The electronic device of any one of claims 10-13, wherein when generating label information of the object sequence image based on the edge segmentation image and the class labeling information, the processor is configured to:

perform morphological transformation and edge segmentation line thickness uniformization processing on the edge segmentation image to obtain an optimized edge segmentation image; and

generate the label information of the object sequence image according to the class labeling information and the optimized edge segmentation image.

15. The electronic device of claim 14, wherein when performing the morphological transformation and the edge segmentation line thickness uniformization processing on the edge segmentation image to obtain the optimized edge segmentation image, the processor is configured to:

perform denoising processing on the edge segmentation image using a first morphological operation to obtain a denoised edge segmentation image;

perform dilation processing on the denoised edge segmentation image using a second morphological operation to obtain an edge segmentation image with continuous segmentation line;

perform erosion processing on the edge segmentation image with continuous segmentation line using a third morphological operation to obtain a transformed edge segmentation image; and

perform skeletonization processing on the transformed edge segmentation image to uniformize a thickness of a segmentation line to obtain the optimized edge segmentation image.

16. The electronic device of any one of claims 10-15, wherein the edge segmentation image comprises a mask image representing edge information of the each object, and/or, the edge segmentation image is the same as the object sequence image in size.

17. The electronic device of any one of claims 10-16, wherein the processor is further configured to:

train an edge detection model based on a first object sequence sample image comprising edge segmentation information of the each object forming the first object sequence sample image.

18. The electronic device of any one of claims 10-17, wherein the processor is further configured to:

train an object sequence recognition model by taking the object sequence image comprising the label information as a second object sequence sample image, the object sequence recognition model being configured to recognize an object sequence based on an image.

19. A computer-readable storage medium having stored therein a computer program, wherein when executed by a processor, the computer program is configured to:

acquire an object sequence image comprising at least two objects arranged in sequence;

perform edge detection on the object sequence image using a trained edge detection

model to obtain an edge segmentation image corresponding to the object sequence image, the edge segmentation image being configured to represent position information of an edge of each object in the object sequence image;

acquire class labeling information corresponding to the object sequence image, the class labeling information comprising class information of the each object in the object sequence image; and

generate label information of the object sequence image based on the edge segmentation image and the class labeling information, the label information comprising class information of pixels in a position region where the each object is located in the object sequence image.

20. A computer program, comprising computer instructions executable by an electronic device, wherein when executed by a processor in the electronic device, the computer instructions are configured to:

acquire an object sequence image comprising at least two objects arranged in sequence;

perform edge detection on the object sequence image using a trained edge detection model to obtain an edge segmentation image corresponding to the object sequence image, the edge segmentation image being configured to represent position information of an edge of each object in the object sequence image;

acquire class labeling information corresponding to the object sequence image, the class labeling information comprising class information of the each object in the object sequence image; and

generate label information of the object sequence image based on the edge segmentation image and the class labeling information, the label information comprising class information of pixels in a position region where the each object is located in the object sequence image.