Ministry of Education of the Republic of Belarus
Educational institution
Belarusian State University of
Informatics and Radioelectronics

UDC 004.75

GAO
Haoxuan

ALGORITHMS OF OBJECTS DETECTION ON A VIDEO SEQUENCE IN DISTRIBUTED SYSTEM

Abstract
for a Master’s Degree
in the Specialty 1-45 80 01 Infocommunication Systems and Networks

____

Supervisor
PhD, Associate professor
O. Shauchuk

____

Minsk 2023

INTRODUCTION

With the explosion of video data close to people's daily life, such as intelligent video surveillance and automatic driving, the realization of video object detection function has changed from a stand-alone system to a distributed system, which has greater application value.

In the distributed system, the realization of video object detection involves network data transmission, and network data transmission is divided into video transmission and image transmission. As for which network data transmission method is more appropriate to use in which scenario, there is no specific conclusion on such a question.

Therefore, the thesis studies and analyzes the applicable scenarios of different network data transmission methods used to realize video object detection in the distributed system.

In order to achieve the goal, it is necessary to solve the following tasks:

– Research and analysis of client-server architecture in distributed system;

– Research on video object detection algorithm;

– Develop the module for interaction between browser application and server (video transmission), develop the module for interaction between Android application and server (image transmission);

– Test the function of the developed program, the network traffic status and the time it takes to complete the video object detection task;

– Analyze the respective applicable scenarios of the browser application and the Android application.

GENERAL DESCRIPTION OF WORK

Relevance of the subject

The work corresponds to paragraph *1 «Digital information and communication and interdisciplinary technologies, production based on them»* of the State Program of innovative development of the Republic of Belarus for 2021–2025.

The work was carried out in the educational institution Belarusian State University of Informatics and Radioelectronics within the framework of research work 21-2033 "Processing, coding and transmission of information in network-centric systems".

The aim and tasks of the work

The aim of the work is to study and analyze the applicable scenarios of different network data transmission methods used to realize video object detection in the distributed system.

To achieve this aim, the following tasks were solved in the dissertation:

1. Research and analysis of client-server architecture in distributed system.
2. Research on object detection algorithm.
3. Develop the module for interaction between browser application and server (video transmission), develop the module for interaction between Android application and server (image transmission).
4. Test the function of the developed program, the network traffic status and the time it takes to complete the video object detection task.
5. Analyze the respective applicable scenarios of the browser application and the Android application.

Personal contribution of the author

The content of the dissertation reflects the personal contribution of the author. It consists in the scientific substantiation of applicable scenarios of different network data transmission methods used to realize video object detection in the distributed system, setting and conducting experiments to study characteristics, assessing the efficiency of it, processing and analyzing the obtained results, formulation of conclusions.

Task setting and discussion of the results were carried out together with the supervisor *Shevchuk O.G, Ph.D., Associate Professor.*

Testing and implementation of results

The main provisions and results of the dissertation work were reported and discussed at: 58th scientific conference of postgraduates, undergraduates and students, (Minsk, April 18–22, 2022) and International scientific and technical seminar "Technologies of information transmission and processing" (Minsk, March - April 2023)

Author's publications

According to the results of the research presented in the dissertation, 2 author's works was published, including: 2 articles and abstracts in conference proceedings.

Structure and size of the work

The dissertation work consists of introduction, general description of the work, four chapters with conclusions for each chapter, conclusion, bibliography, eight appendixes.

The total amount of the thesis is 93 pages, of which 39 pages of text, 34 figures on 16 pages, 7 tables on 3 pages, a list of used bibliographic sources (15 titles on 1 page), a list of the author's publications on the subject of the thesis (2 titles on 1 page), 3 appendixes on 23 pages, graphic material on 10 pages.

Plagiarism

An examination of the dissertation «Algorithms of objects detection on a video sequence in distributed system» by Gao Haoxuan was carried out for the correctness of the use of borrowed materials using the network resource «Bigan» (access address: https://www.bigan.net/) in the on-line mode 31.05.2023. As a result of the verification, the correctness of the use of borrowed materials was established (the originality of the thesis is 94.1%)

SUMMARY OF WORK

The introduction states the aim of the work and gives the tasks were solved in the dissertation in order to achieve the aim.

In the first chapter, the distributed system is introduced, and the client-server model and various client-server cases in distributed system are illustrated. The two cases of client-server are used to develop browser application and Android application, respectively, to analyze the difference between video transmission and image transmission.

In the second chapter, the object detection algorithm YOLOv3-spp is introduced. The part of YOLOX that improves YOLOv3-spp is described. At the same time, the method of using YOLOX to realize object detection is explained. The process of image object detection mainly includes the following steps:

1 Image transformation and normalization

The purpose of transforming and normalizing the images is to obtain images that satisfies the input format of the YOLOX model. The YOLOX model requires the size of the input image to be 640×640 pixels. According to the affine transformation theory, image transformation can be done with the help of transformation matrix M. For image normalization, using the mean and std of ImageNet is a common practice. All models expect input images normalized in the same way, mini-batches of 3-channel RGB images of shape (3×H×W), where H and W are expected to be at least 224. The images have to be loaded in to a range of [0, 1] and then normalized using mean with [0,485; 0,456;0,406] and std with [0,229; 0,224; 0,225].

2 Decode outputs and generate proposals

The output of the model is the tensor of 8400×85 elements, meaning that the model predicts 8400 prediction boxes and every prediction box has 85 properties. Among the 85 properties, the first five properties represent the horizontal and vertical coordinates of the center point of the prediction box, the length and width of the prediction box, and the probability of objects in the prediction box. The remaining 80 attributes represent the object category probability (the model can judge 80 object categories, and each object category will correspond to a probability).

For an image, there are often only a few objects that need to be identified. However, there are 8400 boxes based on the output data. Obviously, the prediction boxes need to be further selected. In this step, it is necessary to manually set threshold to generate proposals with more accurate prediction boxes.

Before manually setting the threshold, it is necessary to know that the output 8400 prediction boxes are merged from three different sizes of anchors. Because in the decoding process, the coordinates of the prediction boxes are related to the corresponding anchor. Among the 8400 prediction boxes, the anchor size corresponding to 6400 prediction boxes is 8×8. This means that the original image of size 640×640 is divided into 80×80 anchors with stride of 8. Similarly, the anchor size corresponding to 1600 prediction boxes is 16×16, and the anchor size corresponding to 400 prediction boxes is 32×32.

Through the anchor, the coordinates of the prediction box can be obtained, and the probability threshold is further set manually. Here the probability threshold is set to 0,6. Then the confidence that each prediction box has the corresponding category of objects can be given:

$$confidence = p * \max(P_c) \tag{2.9}$$

where p – the probability that there is an object in the predicted box;

P_c – the probability of being that class;

Then if the confidence is greater than 0,6, save the proposal (the coordinates of the prediction box) and the corresponding the confidence. After generating the proposals, the possibly correct prediction boxes will be saved.

3 Perform non-maximum suppression

The purpose of performing the non-maximum suppression algorithm is to eliminate redundant overlapping boxes with lower confidences.

4 Accomplish object detection

According to the affine transformation theory, based on the transformation matrix M^{-1} from Image transformation, the coordinates of the final boxes can be converted to the coordinates in the original image. It accomplishes the object detection on the original image.

In the third chapter, the design of server, browser application and Android application are introduced. The developed system architecture diagram is shown in Figure 1, including Server 1, Server 2, Browser and Android.

Server 1 provides API for uploading and downloading video files to process video data sent by the browser application. The main functions realized by Server 1 are video reception, storage and transmission. It also includes reading video frames, using YOLOX to perform object detection on video frames, merging video frames, and saving video information to the database. Server 2 provides API for uploading and downloading images to process the images sent by the Android application. The main function of Server 2 is to use YOLOX to detect the received images and return the detected images.

Browser includes website logic code and graphical interface, where the logic code is responsible for processing network requests and responses, and dynamically updating the graphical interface; the graphical interface is responsible for presenting contents and receiving user operation events.

Android includes data processing module and user interface. For data processing module, it can be represented as two blocks: an interface for interacting with the server API and code for processing the presentation logic. The tool for interacting with the server API is an implementation of the Retrofit interface, which specifies possible HTTP requests that implement the functions of sending and receiving data. The code for processing logic mainly includes reading video frames, sending video frames and merging video frames. For user interface, it consists of two elements: presentation interfaces - an abstract way of interacting with the presentation logic processing classes and graphical user representations - these are objects of the Activity class and XML files.

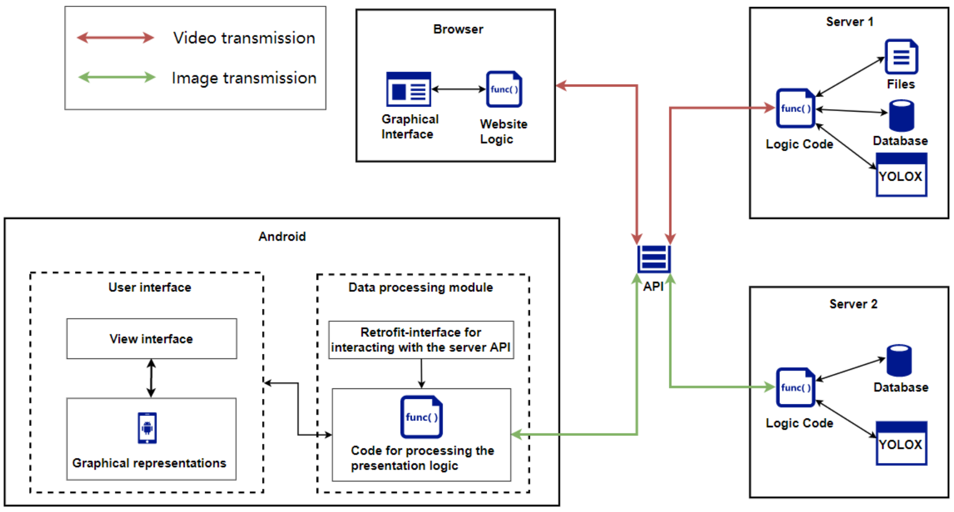

Figure 1 – System architecture diagram

The communication process of Browser application is shown in Figure 2 described as follows.

The user opens the webpage through the browser, and selects a local file to upload according to the prompt on the page. The webpage will judge whether it is a video according to the file type selected by the user. If it is not a video, the webpage will return a prompt that the uploaded file type is wrong; otherwise, upload the video file to the server. Video file is uploaded in multipart/form-data format.

The server receives and saves the video file in the multipart/form-data format, and returns a message that the uploaded video is successful and the file name of the video to the browser webpage. The browser webpage receives a response message that the video upload is successful and displays it to the user, and caches the video file name at the same time.

The user clicks the button of "Detect and Download" according to the prompt of uploading the video successfully. The browser uses the cached video file name and uses the multipart/form-data format to request the corresponding interface of the server. The server receives the request, and starts object detection on the saved video according to the video file name. The processing process includes reading video frames, object detection of video frame (setting a fixed interval, and performing object detection every fixed interval), merging video frames and saving the detected video file. After the task of video object detection is completed, the server returns the meta information of the video, the time of video detection, and the name of video file after detection. And at the same time, executes an asynchronous task to store the information into the database. After the browser receives the response, the web page displays the meta information and detection time of the video, and uses the HTTP GET method to request the video resource through the cached detected video file name. The server receives the request and responds with the video in the form of Blob. Then the browser starts to download the detected video. After the download is successful, the communication ends.

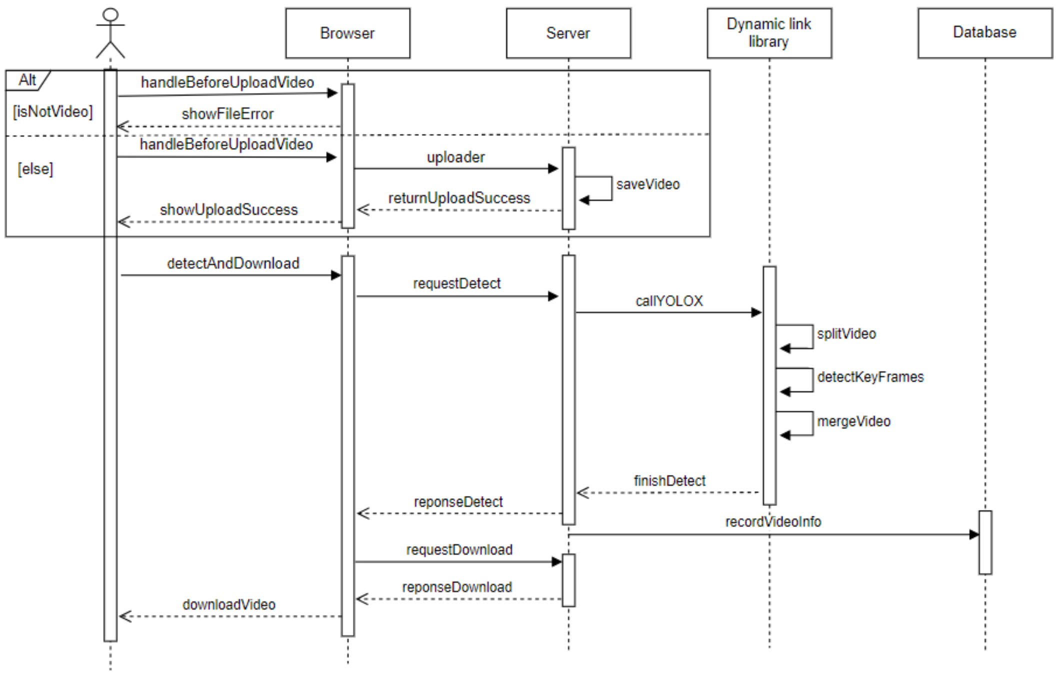

Figure 2 – Sequence diagram of Browser application

The communication process of Android application is shown in Figure 2 described as follows.

The user opens the Android application, clicks the button according to the prompt, and the Android application displays the video file on the device. After the user selects a local video file, the user clicks the button according to the prompt to start processing the video.

The Android application opens the video according to the video file path, and then a loop starts to read the video frames, select the video frames and send them to the server for processing and merging the video frames.

Before starting the loop, set a fixed interval, and process the currently read video frame every fixed interval, while other video frames skip processing and merge directly. The processing process is to cache the video frame in JPEG lossy compression mode and send it to the server in multipart/form-data format. The server receives the image in multipart/form-data format, and converts the received data into Base64-encoded string. Then use the string as a parameter to call the method of the dynamic link library.

In the dynamic link library, the received Base64-encoded string is decoded and converted into Mat type data, and then YOLOX is called for object detection. The dynamic link library converts the detected Mat type data into a Base64-encoded string and returns it. The server returns the Base64-encoded string to the Android application. The Android application decodes it and replaces the image in the cache. Add the replaced image to the task of merging video frames in the loop.

After merging video frames, the Android application saves the processed video, and displays the meta information of the video and the interaction time of the network on the interface. The user can click the play button to play and watch the detected video.

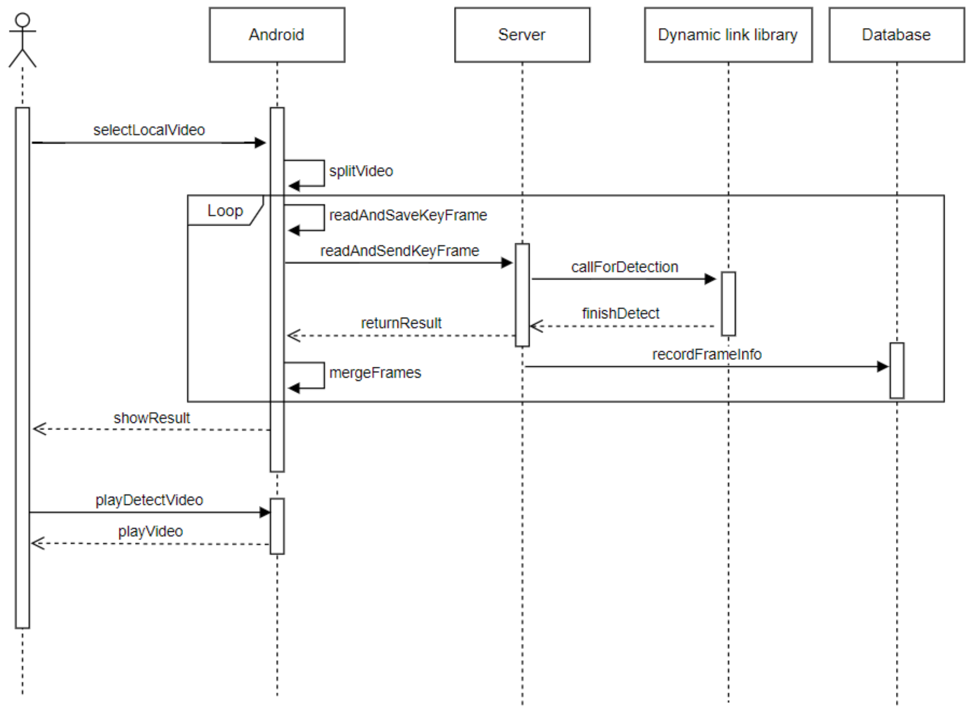

Figure 3 – Sequence diagram of Android application

In the fourth chapter, the functionality of the developed program is tested. In the real network environment, the network traffic status and actual time of video object detection in the distributed system are compared, and the experimental results are analyzed. The video information of the test is shown in Table 1.

Table 1 – Metadata

Video ID	Resolution, pixel	Frame Rate, FPS	Size, MB
1	640 × 360	25	5.07
2	640 × 360	25	23
3	960 × 540	25	8.69
4	960 × 540	25	36.90
5	1280 × 720	25	10
6	1280 × 720	25	61.74
7	1920 × 1080	25	53.66
8	1920 × 1080	25	272.24

When testing the video, the total input and output traffic counted by the server is shown in Table 2.

Table 2 – Server traffic statistics

Video ID	Input Total Traffic for Browser, MB	Output total traffic for Browser, MB	Input total traffic for Android, MB	Output total traffic for Android, MB	Input (Browser)/ Input (Android)	Output (Browser)/ Output (Android)
1	5.279	4.937	0.362	1.161	14.58	4.25
2	23.868	21.853	1.714	4.650	13.92	4.69
3	9.040	8.202	0.608	1.951	14.86	4.20
4	38.573	35.171	2.788	8.016	13.83	4.38
5	13.552	12.173	0.923	2.675	14.68	4.55
6	64.156	57.665	4.459	12.457	14.38	4.62
7	56.043	45.618	2.329	6.968	24.06	6.54
8	284.818	232.499	11.528	34.156	24.70	6.80

It can be seen from the Table 2 that the server processes the same video, but the input and output traffic of different applications are different. Because the browser application transmits video, while the Android application transmits images, the input and output traffic of the browser application is much higher than that of the Android application. Specifically, for videos with resolutions of 640×360, 960×540 and

1280×720, the input traffic of browser application is 13 to 15 times larger than that of Android application, and the output traffic of browser application is 4 to 5 times larger than that of Android application. For videos with a resolution of 1920×1080, the input traffic of browser application is 24 to 25 times larger than the input traffic of Android application, and the output traffic of browser application is 6 to 7 times larger than the output traffic of Android application.

From an economic point of view, for Android application, the billing method of cloud servers based on traffic can be chosen. Pay-by-flow refers to paying for the actual traffic used per hour during the use of the cloud server. It is a pay-after-use model, which is suitable for low-bandwidth usage but intermittent network access peaks scene. For browser applications, the server recommends selecting billing by bandwidth.

As shown in Figure 4, it shows the traffic status of the server when performing video object detection on Video 1-8 under the browser application. The horizontal axis in the figure represents time, and the vertical axis represents the number of bytes (Megabyte). And "In" represents the input traffic of the server, and "Out" represents the output traffic of the server. As shown in Figure 5, it shows the traffic status of the server when performing video object detection on Video 1-8 under the Android application. The horizontal axis in the figure represents time, and the vertical axis represents the number of bytes (Kilobyte). And "In" represents the input traffic of the server, and "Out" represents the output traffic of the server.

From Figure 4 and Figure 5, we can see that the browser application has high requirements on the upstream bandwidth and downstream bandwidth of the server, at least 20M bandwidth is needed to ensure that the video upload and download will not take too much time. However, the Android application has low requirements on the uplink bandwidth and downlink bandwidth of the server, and only needs 2M bandwidth to handle high-resolution video object detection tasks.

As shown in Figure 6, the total time of two applications processing the same video and getting the detected video is showed. In the Figure 6, the horizontal axis represents the ID of the test video, and the vertical axis represents the actual time. It can be seen from the Figure 6 that for low-quality video (640×360, 960×540 and 1280×720) and high-quality (1920×1080) short video, the actual time of Android application is 1 to 1.3 times longer than that of browser application, and for the high-quality (1920×1080) non-short video, the actual time of the Android application is about 0.89 times shorter than that of the browser application. From the perspective of user experience, that is, the waiting time of users, browser applications are more suitable for object detection of low-quality videos and high-quality short videos. For object detection of high-quality non-short videos, Android applications are more suitable.

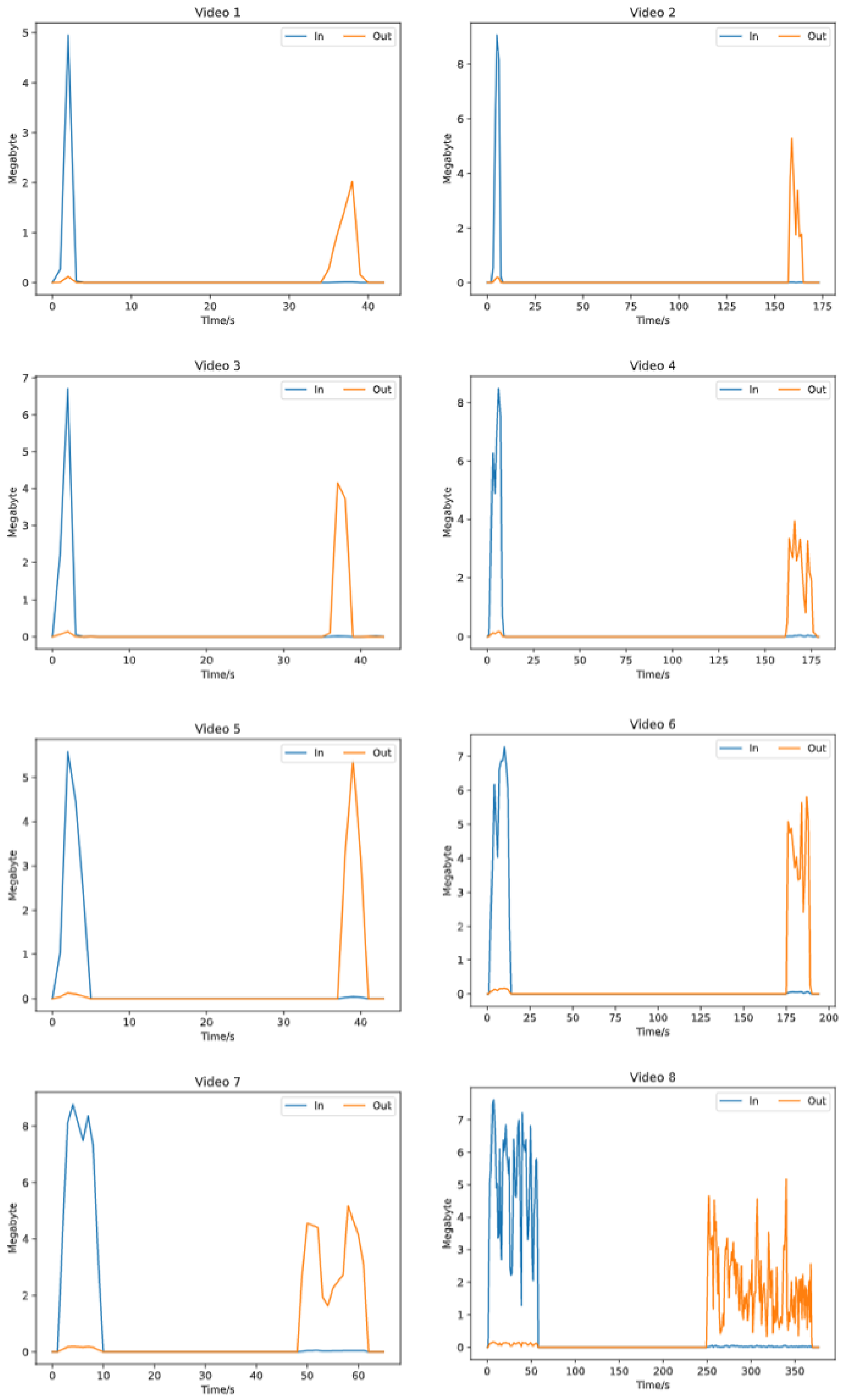

Figure 4– Network data of browser application

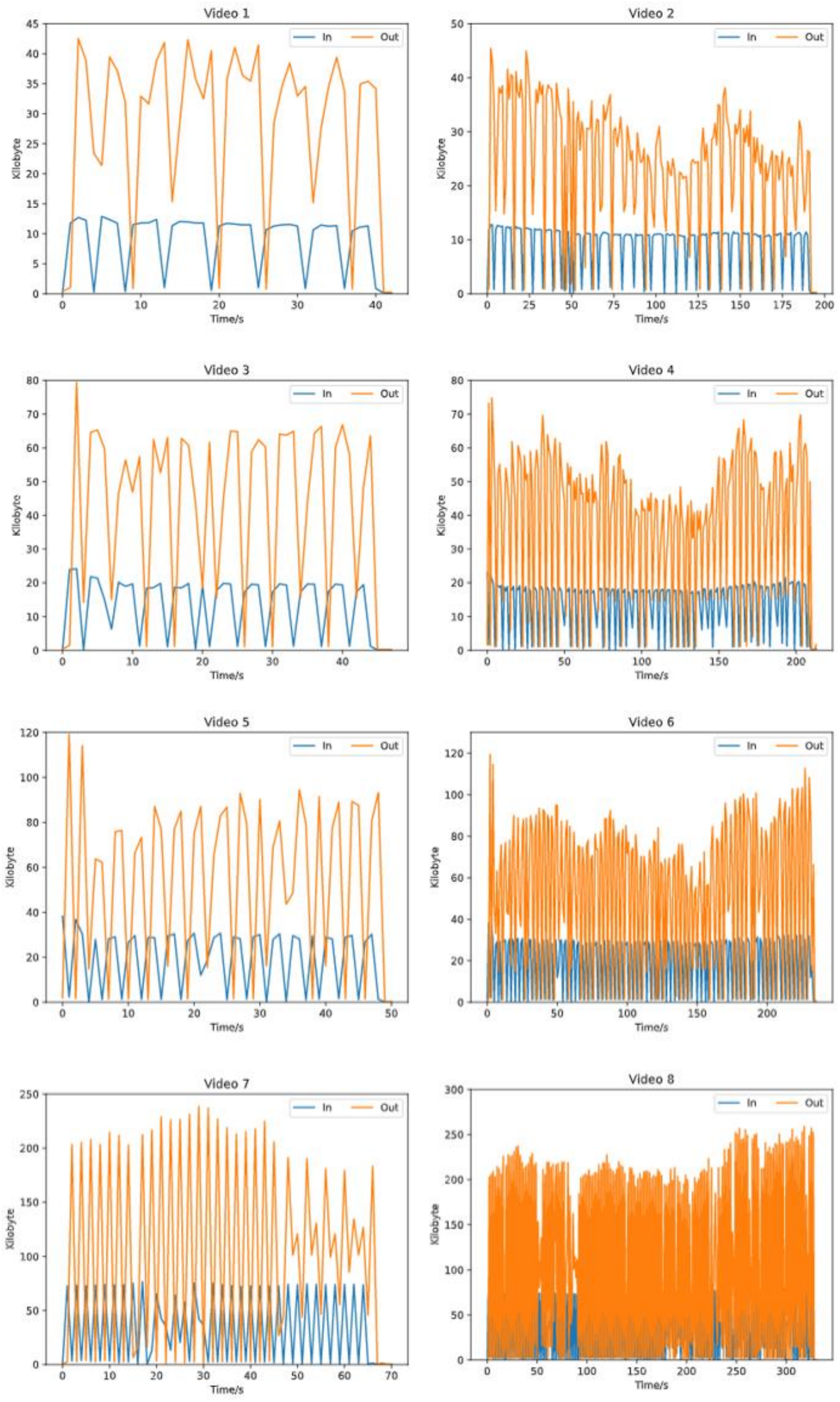

Figure 5 – Network data of Android application

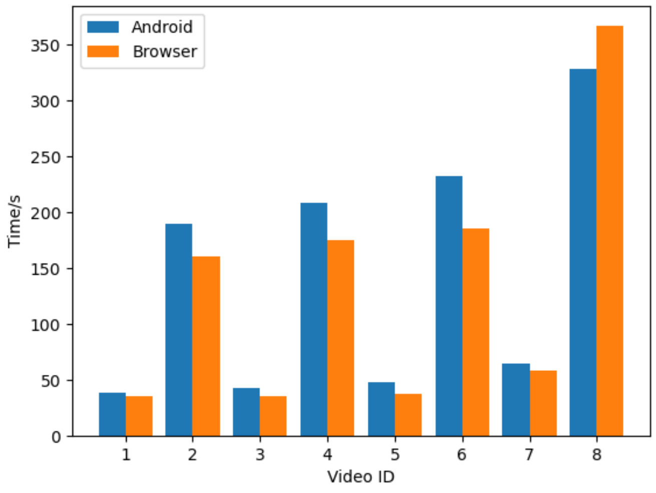

Figure 6 – The total time to complete the detection task

CONCLUSION

In the graduation project, according to different client and server cases in the distributed system, the server, browser application and Android application were developed respectively, and the YOLOX algorithm was applied in the server to realize the video object detection.

The client server cases under the distributed system were introduced specifically. The structure and loss function of YOLOv3-spp were explained. The part of YOLOX improving YOLOv3-spp was explained, and the method of using YOLOX to realize object detection was given. The process of using YOLOX by the server, the structure of the project directory and the process of network data transmission were described in detail. The project directory, interface development and video communication process of the browser application were described in detail. The project directory, interface development and image communication process of the Android application were described in detail.

The functions of the two developed applications were tested in a stand-alone environment, and the network traffic status and actual waiting time were tested in a real network environment. The test results show that the browser application can maintain the stable upload and download of the video, when the server bandwidth is guaranteed to be 20M or higher. At this time, the browser application is more suitable for processing video object detection tasks. When the server bandwidth is within 2M, Android application is more suitable for processing video object detection tasks. For videos with resolutions of 640×360, 960×540, and 1280×720, the input traffic of the browser application is 13 to 15 times that of the Android application, and the output traffic is 4 to 5 times that of the Android application. For a video with a resolution of 1920×1080, the input traffic of the browser application is 24-25 times that of the

Android application, and the output traffic is 6-7 times that of the Android application. From an economic point of view, for browser application, the server is suitable for billing by bandwidth; for Android application, the server is suitable for billing by traffic. From the perspective of user waiting time, for 640×360, 960×540, 1280×720 resolution videos and 1920×1080 resolution short videos, the actual time of Android application is 1 to 1.3 times longer than that of browser application; for non-short videos with the resolution of 1920×1080, the actual time of the Android application is about 0.89 times shorter than that of the browser application. Then browser application is more suitable for processing low-quality videos and high-quality short videos, and Android application is more suitable for processing high-quality non-short videos.

LIST OF AUTHOR'S PUBLICATIONS

A-1 Gao Haoxuan. Object detection on images based on YOLOX / Gao Haoxuan // Инфокоммуникации : сборник тезисов докладов 58-ой научной конференции аспирантов, магистрантов и студентов, Минск, 18–22 апреля 2022 г. / Белорусский государственный университет информатики и радиоэлектроники. – Минск : БГУИР, 2022. – С. 182–187.

A-2 Gao Haoxuan. Video object detection program design under two different client-server organizations / Gao Haoxuan, Shevchuk O.G. // Технологии передачи и обработки информации (Technologies of information transmission and processing): материалы Международного научно-технического семинара, Минск, март – апрель 2023 г. – Минск : БГУИР, 2023. – С. 39–44