Abstract

Surgical workflow analysis is currently an active area of computer vision research, the process of which is to analyze video of endoscopic surgeries to automatically detect its different phases. In this research, the end-to-end recurrent convolutional networks are developed for cataract surgical workflow analysis based on a temporal memory relation network, referred to as CataRCNet. The research goal is to leverage supervised deep learning algorithms to classify cataracts video frames according to their surgical phases. In this project, nine methods are designed to recognize the steps of cataract surgery including three different network architectures with three different backbones, and they all can complete the recognition task. The Non-local operator and temporal variant layer are designed to provide long-range and multi-scale temporal supportive information for accurate classification without disturbing the end-to-end learning process. The highest f1-score of 25 test videos is 0.7796 achieved by the end-to-end ResNet50 plus LSTM model.

Categories: Artificial Intelligence, Computer Vision, Medical Imaging key words: Cataract surgery, Video Analysis, Deep learning, Surgical workflow analysis, Surgical workflow segmentation

Word Count: 9758

Contents

1 Introduction **2**

1.1 Background . . . 2

1.2 Surgical Workflow Analysis . . . 3

1.3 Related Work . . . 5

1.4 Aims and objectives . . . 6

2 Deep Learning for Surgical Workflow Analysis **8**

2.1 Introduction to Neural Networks . . . 8

2.2 Convolutional Neural Network - CNN . . . 9

2.2.1 Motivation of CNN . . . 9

2.2.2 Convolution layers . . . 10

2.2.3 ReLU layer . . . 11

2.2.4 Pooling layer . . . 11

2.3 Sequential modeling . . . 12

2.3.1 Recurrent Neural Network - RNN . . . 12

2.3.2 Lont-Short Term Memory - LSTM . . . 13

3 Dataset - Cataracts **15**

3.1 Introduction of Cataracts Dataset . . . 15

3.2 Preprocessing of Cataracts Dataset . . . 18

4 Methods **19**

4.1 Model Design and Network Architecture - CataRCNet . . . 19

4.1.1 Memory bank: Recurrent Convolutional Network . . . 20

4.1.2 Temporal variation layers: Multi-scale Temporal Kernels . . . 21

4.1.3 Non-local bank operator . . . 22

4.2 Loss Function . . . 23

4.3 Training and Testing . . . 24

5 Experimental Results and Discussions **26**

5.1 Evaluation Metrics . . . 26

5.2 Results and Discussion . . . 28

6 Conclusions and Future Work **37**

A The results of 9 models **43**
A.1 ResNet50 + LSTM 43
A.2 DenseNet169 + LSTM 44
A.3 ResNeSt50 + LSTM 45
A.4 ResNet50+ LSTM + Non-local operator 46
A.5 DenseNet169 + LSTM +Non-local operator 47
A.6 ResNeSt50 + LSTM + Non-local operator 48
A.7 ResNet50+ LSTM + Temporal variation layer + Non-local operator 49
A.8 DenseNet169 + LSTM + Temporal variation layer + Non-local operator 50
A.9 ResNeSt50 + LSTM + Temporal variation layer + Non-local operator 51

Chapter 1

Introduction

1.1 Background

Cataracts occur in the lens of the eye, causing blurred vision as a result of cloudiness [1]. As the lens is usually transparent, it allows light to enter the retina from behind the pupil. The retina transforms this light into a kind of signal in terms of nerves, subsequently sent to the brain where they are interpreted as visual information. The cataract may prevent some of this light from reaching the retina and cause vision to become blurry [1]. In addition to the difficulty of having a clear sight in the dim light, cataracts can also create a halo around bright lights, create a yellow tinge around anything viewed, or fade colors [1].

There is a common phenomenon that a number of human beings suffer from the cataracts, most of which are among the elder aging group. [1]. Globally, cataract is a major health problem as our population ages [2]. An effective treatment for cataracts is only feasible through cataract surgery [2]. Such kind of method is the most effective, with the highest cure rate in the worldwide, thereby 19 million cataract surgeries performed yearly [3]. The cloudy patches affect eyesight gradually because they develop slowly [1]. As a result, the influence of cataracts would extend to the daily life, hindering all the activities that require visual sense, like reading, driving, and etc. To solve this problem, what the surgery does is to replace the affected lens with an artificial one. In cataract surgery, only small incisions are required for the entire procedure. Using high-frequency ultrasound, the natural lens is indeed dispersed into small pieces before leaving the eye [4]. The artificial lens is rolled around an injector before being introduced into the eye. Typically, the cataract surgery takes 15 minutes.

In surgery, surgical microscopes or endoscopes are commonly used for observing the anatomy of the organs. Video analysis of the signals produced by these tools is emerging as a technique for enabling the use of computer-assisted interventions (CAI). A key component of CAI is recognizing what surgical activities your surgeon is performing throughout the surgery [4].

It is currently an active area of computer vision research to analyze video of endoscopic surgeries to automatically detect its different phases[5]. The surgery automation process and the

surgical data science process both rely on this component. For example, the cataract surgery involves a delicate procedure requiring extreme precision, specifically under an ophthalmic microscope. During the procedure, any surgical error may lead to intraoperative complications, which may extend the procedure's time or cause post-operative complications. Through the use of Computer-Assisted Intervention, the surgeon's capabilities can be enhanced. It is not only possible to automate surgical workflow analysis to support surgeons during the procedure, but it can also be used for post-operative analysis of surgical intervention and training junior surgeons as well.

1.2 Surgical Workflow Analysis

The surgical workflow analysis provides a variety of advice for the overall benefits of the patient by ensuring their safety. To be more specific, there are two aspects which such analysis could improve. One is to reduce the uncertainty in the operating process through early warnings of possible errors [6], and the other is to optimize the decision-making process in accordance with the context-aware support [5]. In the real practice, the safety can be guaranteed via adequate preparation prior to surgery, optimizing surgical options with all deviations taken into account [7], and the instant communication during surgery, delivering effective information that is adjusted by accident and change in a timely manner. Traditionally, the video of an operation needs to be manually broken down into stages and annotated with analyzed results, which is a time-consuming work [8]. Due to time constraints, a lot of energy and resources are used to manually search for historical cases, which will leave less time for doctors to make judgments and make surgical plans. For some urgent operations, it will be doomed to give no reference data, leaving a lot of uncertainty. The surgical workflow analysis, however, has the potential to index useful information and monitor possible complications in a very short period of time, if it can be developed [8].

To be specific, the surgical workflow analysis can be applied to the field of robotic surgery. The appropriate use of these computer vision techniques has the potential to improve the safety and efficacy of robotic surgery as it becomes more commonplace [9].With the use of robotics in surgery on the rise, instruments can be tracked in real-time based on their kinematics, enabling more accurate tracking. Based on automated performance metrics, Hung et al. have been using machine learning to assess the performance of different surgeons, recognize surgical activity, and predict surgical outcomes [10]. Robotic systems may also learn to perform surgical tasks using automated performance metrics. It remains to be demonstrated whether robots would be able to operate autonomously even for a limited range of steps [11].

However, some technical difficulties and challenges must be overcome. For all experimental results, it is a great technical difficulty to achieve high accuracy. There are three factors, to some degree, that have an influence on it.

Firstly, some equipment problems can lead to errors in phrase recognition. Because of the fact that the only source of information and data is through video cameras, the quality of the picture will directly affect the analysis and judgment of the data. During the operation, the photogra-

phy is dynamic, causing some blurred images due to focal length adjustment. If this adjustment happens at the key moment of phrase change, the missing information can hardly be recovered from the fuzzy recording. Therefore, the error is positively correlated with photographic failure [8]. In addition, some pollutants, such as blood and fingerprints, can also have a big impact on picture quality. If this is not detected and processed in a timely manner, this video material will be meaningless.

Secondly, as a machine learning program, before being applied into practice, the training and testing process needs a huge number of data. But as an initial correction, this requires labeled annotation and explanation [8]. Therefore, another difficulty is high Labour costs. In addition, in the particularity of the medical field, a lot of data is difficult to obtain because of policy restrictions [12]. As a result, the scarcity of research data will greatly hinder the accuracy of project forecasts.

Thirdly, One root cause is the surgeon's surgical style and experience, which can lead to deviations from standardized surgery [13, 14] . A younger doctor may need to spend more time adjusting and completing each step. Compared with standardized operation, this may increase some meaningless conduction and lengthen the video. For analysis and classification, such data cannot be handled in a way of standard form, but put more pressure on the algorithm. It is not just the quality of the doctor that makes the difference, but also the difference in the patient condition. The severity of the disease directly affects the complexity of the operation [15, 16]. And it's not unusual to spend more time and operations dealing with a relatively serious patient.

1.3 Related Work

In terms of surgical workflow recognition, historically, there have been many researchers in this area of technology, with the achieved phased results. However, such technology still has a long way from maturity. There are a number of technological hurdles to overcome before this technology can be applied to real-world surgery. The researchers who analyzed surgical workflow, uniformly selected surgical videos as the only source for research, because this source is relatively stable in each operation and has minimal impact on the operation [17]. In the selection of video capture methods, scientists mainly focus on two methods. One is to capture the high-level spatial and temporal features. This is because the color, shape and other visual characteristics of each frame will appear differently at different times during the operation [18]. The other is by using auxiliary information to help determine the stage of surgery. For example, by identifying surgical tools, it is often possible to predict what the next action will be, based on their function, and therefore the phase of the surgery.

For surgical phase recognition, Hidden Markov Models (HMMS) were some of the initial approaches, and they captured video information using Dynamic Time Warping (DTW). However, they were extremely limited in application areas [19]. Then EndoNet raised the use of a shared feature extractor and a hierarchical HMM for the surgical tool and phase recognition [8]. Specifically, this method used AlexNet as a feature extractor for tool recognition and a hierarchical Hidden Markov Model (HHMM) for inferring the phase [8]. Additionally, it added the surgical tool information to help support the phase recognition given that it can be proved that the specific tool can only be used to finish the fixed action [20, 21]. But later, the utilization of LSTM replaced the HHMM, whose system also won the MICCAI 2017 EndoVis workflow recognition challenge [22]. Based on this, EndoLSTM was further developed, with a Convolutional Neural Network (CNN) as a feature extractor and an LSTM [23] for feature refinement [19]. For the next step, this method evolved into Endo2N2, which can utilize the prediction of the Remaining Surgery Duration (RSD) to train the feature extractor [24]. Next, both a CNN-LSTM model and SV-RCNet were trained as an end-to-end ResNet and LSTM model for better phase recognition [25, 26].

The hierarchical processing of videos for action segmentation used temporal convolutions [27]. Comparatively to RNNs, an encoder-decoder architecture was capable of capturing both high- and low-level features. TCNs were subsequently fitted with dilated convolutions [28] for action localization and achieved higher performance via greater temporal resolution due to a larger receptive field. A model called Multistage TCNs (MS-TCNs) [29] was introduced for action segmentation and consisted of stages of stacked predictors. During each stage, the initial prediction was incrementally refined with a multilayered TCN.

MTRCNet-CL is the base network utilized by the team that won the first award at MICCAI 2020 EndoVis Challenge [30]. A multi-task problem was approached by MTRCNet-CL [31]. An LSTM model [23] was built with features extracted from the frames to predict tool information along with surgical phase predictions. The two tasks were synergized using a correlation loss. A component of all the above methods is the use of LSTMs, which can only hold in memory short

sequences that are not long enough to span minutes or hours, which are the average duration of a surgery. Due to this, they are unable to infer parallelizably, which would make them useful for integration with an online OR scenario.

The SlowFast Network was applied by two of four teams at MICCAI 2020 EndoVis Challenge, and one team won the third award [30]. The SlowFast Network is designed by Feichtenhofer, et al. for a better recognition of surgical video [32]. This network makes full use of the time axis as it runs at two different forms of frame rates. More specifically, the slow pathway focuses on several specific frames in a large volume of stride video because it values the information of the spatial aspect. In another way, the fast pathway is with high frame rate and high temporal resolution features, but low channel capacity. That is to say, it captures the temporal information within the dynamic video frames. By combining these two kinds of information, it is more supportive to recognize the accurate surgical phase.

Although the technology continues to evolve, current methodologies can only handle short periods of information capture or lack the ability to accurately capture the spatio-temporal characteristics of video. Turning to a new approach, Yueming et al. proposed a temporal memory relation network (TMRNet) to strengthen the long-term information connection within an operation [17]. As a supportive information, this method provides a holistic analysis rather than a piecemeal interpretation. To realize this, TMRNet designed a long-range memory bank, which provides clues about the operation over a long period of time and therefore ensures the features of context. Experimental results show that it is more accurate than the historical method [17].

1.4 Aims and objectives

The main aim of the project is to propose an efficient solution for cataract surgical workflow analysis, which may contribute to post-operative evaluations of the surgical intervention, surgical education programs and real-time decision support. Specifically, the project goal is to develop supervised deep learning algorithms to classify cataracts video frames according to their surgical phases, referred to as surgical phase recognition. The spatial semantics and motion at fine temporal resolution are captured for this project, so it is computationally intensive.

The objectives of this project are:
(1) Understand the surgical workflow analysis.
(2) Understand the fundamental of convolutional neural network and recurrent neural network.
(3) Propose a spatio-temporal model to classify the phases in cataract surgery.
(4) Compare the proposed model with different models.

The contributions of this research:
(1) CataRCNet: The first use of TMRNet for Cataracts surgery.
(2) SV-DenseRCNet: The first use of DenseNet169 as the backbone of TMRNet.

To achieve this goal, this report will firstly introduce fundamental theories related to the deep

learning field, which would be applied into surgical workflow analysis in Chapter 2, and then demonstrate the CATARACTS dataset and the pre-processing strategy, followed by the research methodology and results evaluation in the end.

Chapter 2

Deep Learning for Surgical Workflow Analysis

A lot of approaches for surgical phase recognition are developed based on convolutional neural networks to capture the spatial information and recurrent neural network to model the temporal information [19]. The term Deep Neural Network (DNN) or Deep Learning is reference to the Artificial Neural Networks (ANN) with multi layers [33]. This tool is known as the most powerful network that can deal with huge volume of data. This chapter will introduce the fundamental of Neural Networks, Convolutional Neural Networks, and Recurrent Networks respectively.

2.1 Introduction to Neural Networks

Among the most common models used in machine learning are neural networks, which can be defined using the book of Haykin [34]. Neurons are fundamental units of this model which are defined by Equation 2.1, where w_{kj} represents the synapses or connecting links with k representing the neuron, and j representing the input x_j [34].

$$v_k = \sum_{j=1}^{n} w_{kj} x_j \tag{2.1}$$

$$y_k = \varphi(v_k + b_k) \tag{2.2}$$

Furthermore, bias is yet another element we use to increase or decrease neuron output It is represented by b_k before we apply an activation function. Neuron as a representation can be seen in Figure 2.1, where y_k indicating the output signal of this neuron.

There may be a lot of neurons sharing the same inputs, but connected by different synapses and they are called layers. Multilayer perceptrons consist of an input layer with every unit standing for a signal input, one or more hidden layers where every unit represents a neuron, and an output layer where the network shows the result of processing the input signal, which traverses the network layer-by-layer. This type of network is typically trained using the error back-propagation algorithm. Essentially, it consists of two phases: a forward pass occurring when we receive the

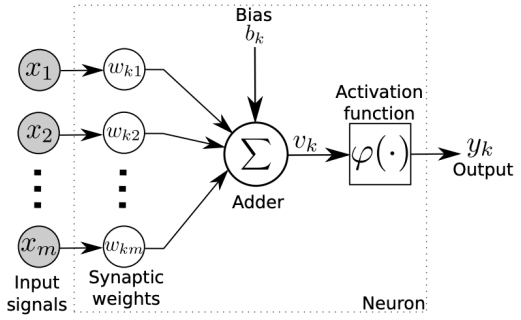


Figure 2.1: The example of a neuron [35]

input signal, which traverses the network until it reaches the output layer, and a backward pass where synaptic weights are adjusted based on an error-correction rule [34].

2.2 Convolutional Neural Network - CNN

The Convolution Neural Network (CNN) is one of the most popular deep neural networks, whose layers includes convolutional, fully-connected, pooling and non-linearity layer [33]. And CNN, to some degree, has some significant influence on the field of machine learning. The deep convolutional network has been widely applied in the fields like image classification data set (Image Net), computer vision, and natural language processing (NLP), where it recognizes the pattern and concepts through the automatic learning and vertical stratification [33].

The Convolution Neural Network is one of the most popular deep neural networks, which include convolutional layers, fully-connected layers, pooling layers and non-linearity layers. An example CNN model: $input \rightarrow convolution- > ReLU \rightarrow maxpooling \rightarrow affine \rightarrow ReLU \rightarrow affine \rightarrow softmax$is shown in Figure 2.2.

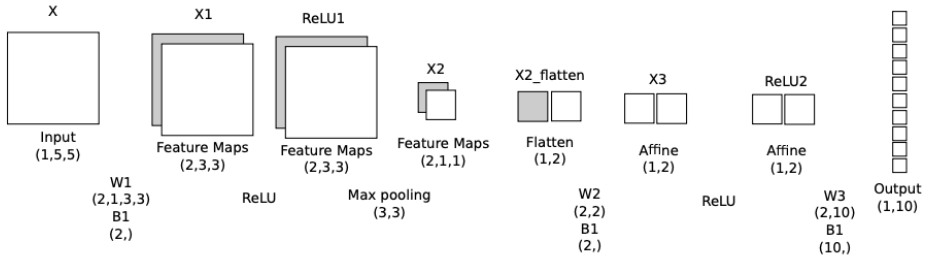


Figure 2.2: An example architecture of CNN [35]

2.2.1 Motivation of CNN

The appearance of CNN is to solve the problems of low efficiency and heavy workload in image data processing of NN [36]. In the traditional processing method, NN correlates the input and

output of each pixel, resulting in a full-coverage workload. For efficiency, CNN raised a concept named Kernel. The aim of Kernel is to store the key features with meaningful and useful information, which is much smaller than the traditional input. For example, the traditional inputs may have thousands or millions of pixels while by contrast, the Kernel only has tens or hundreds of pixels, a hundredfold or more difference in quantity. By comparing kernel parameter values, it can process inputs and produce outputs more quickly, thereby a large improvement in efficiency.

2.2.2 Convolution layers

A 2D convolution is equal to applying a filter to an image in terms of its 2D nature. Rather than using hand-crafted features, it helps us to identify classes of an image dataset using features derived from the CNN [37]. CNNs consist primarily of convolutional layers, and the use of these layers has helped compute the best state-of-the-art results for classifying small data sets (mnist, cifar-10) as well as the most challenging datasets (ImageNet). Presented here is a brief explanation of how convolution works in a 2D image.

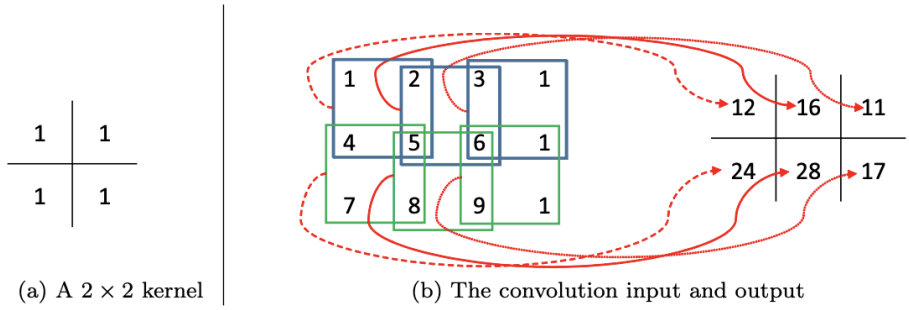
Convolution layers feed forward in a pretty simple way, but they are computationally expensive. A convolution operation is denoted by a " $*$ ", like moving a window over a subregion of the input image, since in the subregion we operate on the multiplication of the elements-wise and sum each result and sometimes add bias, the process is repeated until all the subregions are covered [35].

There are two important concepts to understand convolution operation:
Stride: We move to the right a certain number of spaces before reaching the end of the image, and we move downward a certain number of spaces before we reach the end.
Padding: In some cases, we want to make use of every pixel in the image. As such, padding simply indicates how many columns and rows of zeros will be added to the border of the image. Additionally, a 'full' convolution requires that you add an additional padding with the length of "the width of the filter - 1". As per Wu, the convolution procedure can be expressed as Equation 2.3 in precision mathematics [37].

$$y_{i_{l+1},j_{l+1},d} = \sum_{i=0}^{H} \sum_{j=0}^{W} \sum_{d_l=0}^{D_l} f_{i,j,d_l} \times x^{l}_{i_{l+1}+i,j_{l+1}+j,d_l} \tag{2.3}$$

With a clear understanding of what convolution means for images in the form of height, width, and channels, we can construct some examples to clarify the concept [35]. Also, it's important to know that we can manage the input dimensions in the form $D_l \times H_l \times W_l$, rearranging kernel dimensions $D \times Dl \times H \times W$. With a little parentheses, we get a kernel (also called a filter) that rotates 180 degrees when the convolution is applied.

For instance, it is supposed that the input image is 3×4 and the convolution kernel size is 2×2 as demonstrated in Figure 2.3.



(a) A 2×2 kernel

(b) The convolution input and output

Figure 2.3: An example of convolution operator [37]

2.2.3 ReLU layer

In CNNs, Rectified Linear Unit (ReLU) is the most widely used activation function, which has a good training speed and superior results compared to others such as tanh and sigmoid [35].

Essentially, functions return 0 when the value provided is negative and the value itself in other cases. If input x is given, the output of a ReLU is:

$$y_{output} = max(x, 0) \tag{2.4}$$

2.2.4 Pooling layer

The pooling function, which is at the final stage of a convolutional network, is used to make further modifications for the output of the layer [36]. And the pooling function has specific sub-function names for its specific purposes. For example, the max pooling function is related to the maximum output of a specific rectangular neighborhood. Similarly, other functions include the average pooling function, the weighted average pooling function or others with detailed logics. The max pooling will be illustrated in the next part.

Max-pooling is a non-linear form of down-sampling. The Max-pooling algorithm divides the input image into a set of non-overlapping rectangles, and each of these rectangles is output as the maximum value [35]. Similar to a convolution operation, we simply take the maximum value rather than applying a filter to every sub=region, while we can also take an average pooling (the average value of the subregion) or a Lp-pooling [4]. But in many cases, most architectures use maximum pools, so we will focus on that one.

In the case shown in Figure 2.4, the expected output is a 2×2 image if 1-channel image of size 4×4 is applied by a 2×2 max pooling.

Max-pooling input

23	4	16	90
12	32	12	45
5	7	8	9
2	12	14	56

Max-pooling output

32	90
12	56

Figure 2.4: An example of the max pooling [37]

2.3 Sequential modeling

2.3.1 Recurrent Neural Network - RNN

Recurrent Neural Network (RNN) is a technique for handling sequential data which is with variable length inputs or outputs [36]. Sharing parameters across different parts of a model is leveraged for recurrent networks.

The recurrent circuits can be designed in a wide variety of ways, all of which are compact and easy to understand. The explanation will be given that, the equivalent unfolded graph, which is computationally advantageous to improve focusing on the idea of information flow forward in time (computing outputs and loss) and backward in time (computing gradient), can be automatically obtained. The simplest recurrent network architecture is shown in Figure 2.5.

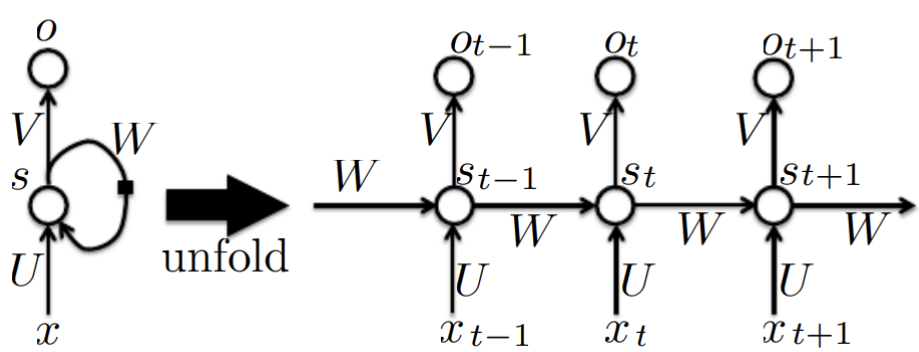


Figure 2.5: An example architecture of RNN [36]

In Figure 2.5, the left structure illustrates that a normal recurrent network circuit model that includes hidden-to-hidden recurrence, with weight matrices for each of the three kinds of connections (input-to-hidden U, hidden-to-output V, hidden-to-hidden W, respectively). Circles indicate the corresponding vector of activations. The right part shows that each node now represents an instance of time like a time-unfolded flow graph.

The corresponding equations for the above recurrent network if the hyperbolic tangent non-

linearities are utilized in the hidden layer and the softmax function is used in the output, are shown in Equations 2.5 - 2.8 [36].

$$\alpha_t = Ws_{t-1} + Ux_t + b \tag{2.5}$$

$$s_t = tanh(\alpha_t) \tag{2.6}$$

$$o_t = c + Vs_t \tag{2.7}$$

$$p_t = softmax(o_t)[36] \tag{2.8}$$

where the parameters U, V, W indicate the weight matrices for input-to-hidden, hidden-to-output, and hidden-to-hidden respectively. In addition, the parameters b, c represent the bias vectors. This architecture is an example of a recurrent neural network which maps an input sequence to an output sequence with the same length.

2.3.2 Lont-Short Term Memory - LSTM

The standard RNN has a drawback that it cannot contain over 5-10 times of steps [38] at the same time due to the fact that the back-propagated error signals cause two kinds of results: either the growth or the shrink of itself every time step done. Therefore, in many cases, the error will be enlarged or disappear [39, 40] over many steps, Error signals that are blown up create oscillating weights, while errors that vanish create learning that takes too long or is unsuccessful.

Long short-term memory (LSTM) is a gradient-based method that solves the vanishing error problem as described in [38, 41] And LSTM, fundamentally, applies constant error carousels (CECs), enforcing a constant error flow within special cells, to solve the problem of bridging minimal time lags of more than 1,000 discrete time steps. Gate units that grant access to cells are multiplicative, which makes decisions depending on how the cells are configured.

Without the newly input of the cell, though, it is known to all that the reflux of the CEC stays unchanged, as one member of the neural network, it is not only connected to itself, but also with the connection of other units in the neural network, with the consideration of additional weighted inputs and outputs [42]. As such, as a result of using the same mass to store and ignore input, neuron u may receive conflicting weight update signals. When a neuron u has a weighted output connection, the same weight can be used to retrieve its content and prevent it from communicating with the rest of the network.

LSTM addresses the conflicting weight update problem by extending the CEC by creating an access for input layer and other memory cells to connect the input and output gates. As a result, we get a more complex LSTM unit, called a memory block, whose standard architecture is shown in Figure 2.6.

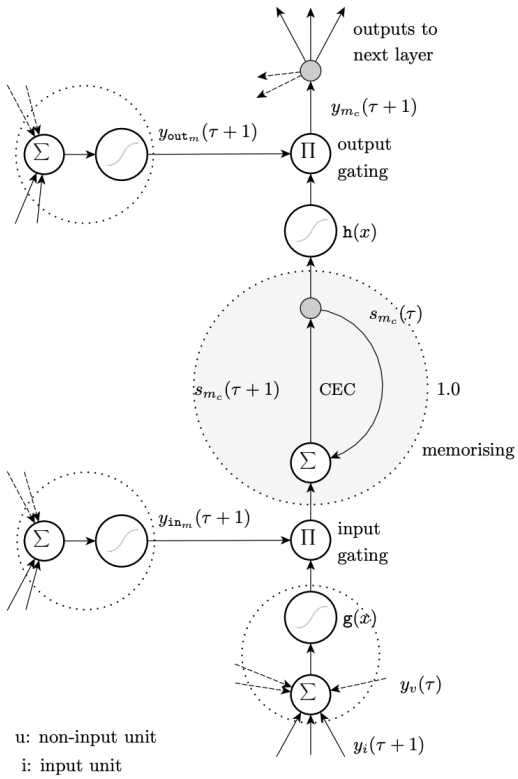


Figure 2.6: A memory block for LSTM [42]

By scaling the signals from the network to the memory cell appropriately, input gates control the signals through sigmoid threshold units consisting of [0, 1]; when the gate is closed, activation levels are close to zero[42]. Moreover, these can learn to function reliably even if they are exposed to irrelevant signals. The definition of the cell state is to describe the activation of a CEC by the input gate. It can be learned how to control the access of the output gates to the contents of the memory cells, which can protect other memory cells from disturbances originating from u. Therefore, multiplicative gate units function primarily by allowing or denying constant error flow between the CEC and the PFU.

Chapter 3

Dataset - Cataracts

CATARACTS is employed in this project, which is a public challenge dataset from Medical Image Computing and Computer Assisted Intervention (MICCAI) 2020 Endoscopic Vision Challenge (EndoVis) [4].

3.1 Introduction of Cataracts Dataset

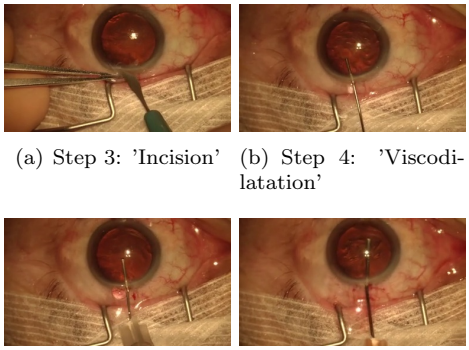
For CATARACTS, a collection of 50 videos representing cataract operations at Brest University Hospital between January 22, 2015 and September 10, 2015 is included in this dataset [43]. Among the reasons for surgery were age-related cataracts, traumatic cataracts, and refractive errors. An average age of 61 years was observed in the patients (minimum: 23, maximum: 83, standard deviation: 10) [43]. Twelve males and 38 females participated. All patients provided informed consent. During the surgery, three surgeons performed it: a renowned expert (48 procedures), a one-year experienced surgeon (1 procedure) and an intern (1 procedure) [43]. The surgeries were performed using an OPMI Lumera T microscope (Carl Zeiss Meditec, Jena, Germany). We used a Mitsubishi 180I camera (Tokyo, Japan) and a MediCap USB200 recorder (Plymouth, USA) when capturing the videos. On average, videos lasted 10 minutes and 56 seconds (minimum: 6 minutes 23 seconds, maximum: 40 minutes 34 seconds, standard deviation: 6 minutes 5 seconds). The total number of recorded surgeries exceeds nine hours. Approximately 30 frames were displayed per second at a resolution of 1920x1080 pixels [43].

These videos are segmented into 19 phases by a non-M.D. expert according to the ones provided in [44]. Each of the videos was done frame-by-frame. In the recorded surgeries, the 19 steps are noted as Table 3.1:

Table 3.1: Dataset Cataracts steps

Step ID	0	1	2	3
Step Name	Idle	Toric Marking	Implant Ejection	Incision
Step ID	4	5	6	7
Step Name	Viscodilatation	Capsulorhexis	Hydrodissetion	Nucleus Breaking
Step ID	8	9	10	11
Step Name	Phacoemulsification	Vitrectomy Marking	Irrigation/Aspiration	Preparing Implant
Step ID	12	13	14	15
Step Name	Manual Aspiration	Implantation	Positioning	OVD Aspiration
Step ID	16	17	18	
Step Name	Suturing	Sealing Control	Wound Hydratation	

The example frames of "train01.mp4" in step 3 ,4 ,5, 6 are shown in Figure 3.1.



(a) Step 3: 'Incision' (b) Step 4: 'Viscodilatation'

(c) Step 5: 'Capsulorhexis' (d) Step 6: 'Hydrodissetion'

Figure 3.1: Example frames of step 3, 4, 5, 6

The ground truth files are given in the form of CSV files. A CSV file is associated with each video, containing a row for every frame along with a header. In the first column is the frame index. A frame's step index is displayed in the second column. A sample annotation file looks like Figure 3.2:

train01

Frame	Steps
1	0
2	0
3	0
4	0
5	0

Figure 3.2: The sample annotation csv file

The Cataracts dataset consists of 50 videos recording the cataract surgery procedures, including training dataset from "train01.mp4" to "train25.mp4" and test dataset from "test01.mp4" to "test25.mp4". The ground truth files illustrate that not all surgical videos have all steps from step 0 to step 18. All videos have step 0, 3, 4, 5, 6, 7, 8, 10, 13, 14, 15, and 18, referred to as the common steps. Additionally, the special steps are provided following the rule that the steps are not commonly shared with other files, with the unique characteristics, including step 1, 2, 9, 11, 12, 16, 17. The more detailed information of the 25 training videos is summarized in the Table 3.2.

Table 3.2: Cataracts: the ground truth files information

Ground truth files	Number of frames	With Steps	With Special Steps
Train01	14380	0, 3-8, 10, 11, 13-15, 18	11
Train02	11837	0, 3-8, 10, 13-15, 18	-
Train03	17490	0, 3-8,10, 13-15, 18	-
Train04	13266	0, 3-8,10, 13-15, 18	-
Train05	12778	0, 3-8,10, 13-15, 18	-
Train06	13848	0, 3-8,10, 13-15, 18	-
Train07	18769	0, 3-8,10, 13-15, 18	-
Train08	27463	0, 3-8,10, 13-15, 18	-
Train09	15973	0, 3-8,10, 13-15, 18	-
Train10	25569	0, 3-8,10, 13-15, 18,17	17
Train11	14252	0, 3-8,10, 11, 13-15, 18	11
Train12	21574	0, 2,3-8,10, 13-15, 18,16	2, 16
Train13	14021	0, 3-8,10, 11, 13-15, 18	11
Train14	20544	0, 2,3-8,10, 13-15, 18,16	2, 16
Train15	13762	0, 3-8,10, 11, 13-15, 18	11
Train16	18232	0, 3-8,10, 11, 13-15, 18	11
Train17	12433	0, 3-8,10, 11, 13-15, 18	11
Train18	13952	0, 3-8,10, 11, 13-15, 18	11
Train19	71066	0, 3-14, 16-18	9, 11, 12, 16, 17
Train20	16939	0, 3-8,10, 11, 13-15, 18	11
Train21	15463	0, 3-8,10, 13-15, 18,16	16
Train22	20664	0, 3-8,10, 13-15, 18	16
Train23	11182	0, 3-8,10, 13-15, 18	16
Train24	18591	0, 1, 3-8, 10, 11, 13-15, 18	1, 11
Train25	40820	0, 3-8, 10, 13-18	16, 17

Test → X1, 7, 14, 16 19

6 - 25 → Train
1 - 5 → Validation

3.2 Preprocessing of Cataracts Dataset

According to Table 3.2, step 1 only shows in "train24.mp4" video, and step 9 only shows in "train19.mp4" video. In this project, 25 "train" videos are divided into 20 videos for training and 5 videos for validation. 20 Videos from "Train06.mp4" to "train25.mp4" have more special steps, so they are chosen as the training set. In addition, 5 Videos from "train01.mp4" to "train05.mp4" are selected as validation set. For comparison, the 20 test videos are the same as those in benchmark challenge (MICCAI EndoVis Challenge 2020), which are from "test01" to "test25" except "test01", "test07", "test14", "test16", "test19".

The original frame rate of Cataracts is about 30 fps, and each frame has an original resolution of 1920×1080. Firstly, the resolution of videos is changed to 384×216 using ffmpeg. Then, the videos are converted to frames, and the frame rate is down sampled to about 1fps. These two steps are used to save memory and reduce the computation cost. For data augmentation, the train dataset is transformed by 216×216 cropping, color Jitter, horizontal flip and rotation.

Chapter 4

Methods

4.1 Model Design and Network Architecture - CataRCNet

This research is based on an end-to-end temporal memory relation network (TMRNet) proposed by Jin et al. [17], and the network is implemented with the benchmark Cataracts dataset, which is called Cataract Recurrent Convolutional Network (CataRCNet). It is significant to capture the long range temporal dynamics effectively for workflow recognition problem. The developed network can collect long term context.

For cataract surgical workflow recognition, CataRCNet is an end-to-end recurrent convolutional network for relating long-range and multi-scale temporal patterns which are utilized to augment the present features. This network includes a long-range memory bank, temporal variation layers, a non-local operator. Firstly, a long-range memory bank is established using a recurrent convolutional network to store the rich supportive information. Then the memory bank information is through multi-scale temporal-only convolutions for capturing complex multi-scale temporal patterns. For end-to-end learning, a non-local operator is introduced to relate the past to the present. The CataRCNet architecture is presented in Figure 4.1.

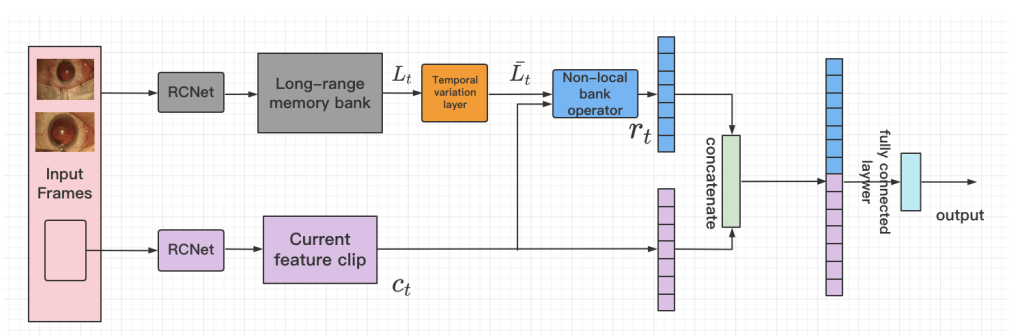


Figure 4.1: The CataRCNet architecture for surgical workflow recognition

Figure 4.1 demonstrates that the long range information is stored by the long range memory bank firstly, and then it is through the temporal variation layer for tolerating the multi-scale temporal cues. The long range features L_t for frame x_t from the memory bank LB are selected

Doubt

to augment current frame feature c_t. Then they are through temporal variation layer to increase the multi-scale temporal extent. Finally, non-local operator is used to relate the current feature to long range temporal dependency. Three mentioned blocks will be deeply illustrated in the following part.

4.1.1 Memory bank: Recurrent Convolutional Network

The Recurrent Convolutional network is leveraged to build the long range memory bank which is helpful to obtain the long complex context. In order for a model to make accurate prediction, it is beneficial to relate distant cues in time to the present state of affairs. In place of relying on the online training such as 3D Convolutional models or Long Short-Term Memory with short-term view, the CataRCNet has a long range memory bank (LB) to store the temporal information of the entire video. This long-range memory bank servers as an auxiliary memory cell to recognize the workflow phase at this point in time. When training the network, the gradients and back propagation of the memory bank are not required, so the model can be trained with the end-to-end learning.

The unified recurrent convolutional network (SV-RCNet) [25] is selected to build the memory bank. In this model, a fifty-layer residual network (ResNet50) is integrated with an LSTM network seamlessly. A more powerful state-of-the-art classification model ResNeSt50 [45] is also used to establish the CataRCNet for comparison.

In addition, another model is proposed to build the memory bank, and it is advanced by the seamless integration of Dense Convolutional Network 169 (DenseNet169) [46] with an LSTM network. It is the first time that the DenseNet169 is designed as the backbone of TMRNet. This convolutional model offers several compelling benefits including mitigating the vanishing gradient problem, making features propagate more efficiently, encouraging feature reuse, and reducing the number of parameters significantly [46]. The proposed network, referred to as SV-DenseRCNet, is shown in Figure 4.2.

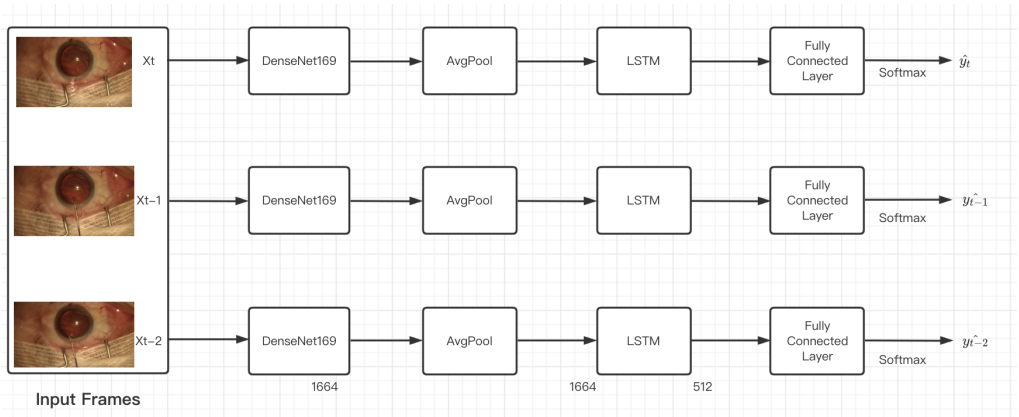


Figure 4.2: SV-DenseRCNet: DenseNet169 with LSTM based on SV-RCNet

As depicted as Figure 4.2, the SV-DenseRCN includes the DenseNet169, which needs to be

deeply used to distinguish the visual features, and the LSTM network, which catches the temporal information via sequential frames. By integrating the two parts above, the end-to-end recurrent convolutional network is formed and it would be with much help in dealing with the recognition problem. The inputs of the whole network are video clips for obtaining spatial and temporal information efficiently. There is always a difficulty that two frames in different phases may share very similar features. This phenomenon adds many obstacles to the discrimination process. Therefore, the temporal information fundamentally solves this problem as it will automatically exclude phases that have ended in the time series. And the LSTM is exploited to organize time information for better judgment. For SV-DenseRCNet, the visual features extraction relies on a set of dense convolutional blocks. Deep decisions are made to mine distinguishing characteristics through the input and output of these successive blocks. And this approach could ease the difficulty of complicated underlying transformation. Although the visual feature extraction and LSTM network respectively have the video recognition, their results need to be processed by another process to achieve compatibility and deep classification. For the integration, SV-DenseRCNet was trained in an end-to-end manner to optimizing the parameters of DenseNet and LSTM network. To be specific, this network divides the whole video into several clips to utilize both temporal and visual information rather than the single frame. In addition, this way of processing could alleviate the training difficulty and the limitation of computational memory.

The long-range memory bank **LB** is in the form of $\{l_0, l_1, ..., l_{T-1}\}$, a time indexed feature list for a T-length video with time steps of $[0, 1, ..., T-1]$. In particular, the feature l_t of frame x_t in the memory bank is generated by extracting a video clip $x = \{x_{t-m}, ..., x_{t-1}, x_t\}$ as the input, where x_t represents the current frame of time t. The model will output a high-level spatio-temporal feature vector l_t with the dimension of 512. Then the clip-length m temporal cues can be embedded into the current frame feature. The time index is also recorded into the memory bank, and it is helpful to locate the corresponding feature clip for information augmentation. Mech

4.1.2 Temporal variation layers: Multi-scale Temporal Kernels

In temporal duration, the actions in the same phase might be with high variability. Using the fixed kernel size to model the temporal dynamics is limited to analyse the surgical workflow, and the multi-scale kernels of temporal variation layer are beneficial to tolerate the varieties in temporal extents. The specific architecture of the temporal variation layer is presented in Figure 4.3.

As shown in Figure 4.3, the temporal variation layer contains five operators, and its input is the long-range feature $L_t \in R^{L\times512}$ where L indicates the length step. Three of them are convolutional layers with different kernel sizes. Then a max pooling operation is executed with the stride of 1, and kernels size of 2. The fifth operation is just to connect the input features. After these layers, five feature vectors are obtained, and they will be fused by concatenating. Next, the output features need to be through a max pooling layer with kernel size of 5. Finally, the output of temporal variation layer $\bar{L}_t \in R^{L\times512}$ is with the same dimension as the input. The long-term supportive information is augmented by encoding features with multiple scale kernels of convolutional layers. Therefore, the representation capacity of long-range features can be enhanced by the temporal variation layer.

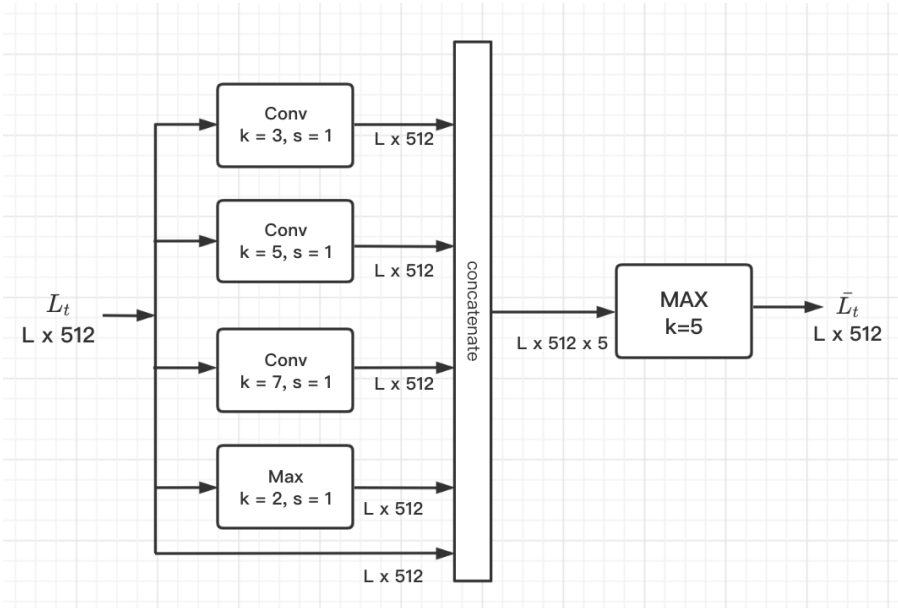


Figure 4.3: Temporal variations layers

4.1.3 Non-local bank operator

The temporal memory relation network relates the temporal cues using a non-local operator for leveraging the long-range and multi-scale past temporal information effectively [47]. Non-local operations presented here are an efficient, simple, and generic approach to capture long-term dependencies. Non-local operation offers the follow advantages: this algorithm accounts for long-range dependencies directly by calculating the interactions between two points; it can achieve good results with only a few layers; it can combine different sizes of input easily [47].

Firstly, the recurrent convolutional network is employed to extract the feature c_t of the current frame x_t by inputting the local short video clip $x = \{x_{t-n}, ..., x_{t-1}, x_t\}$. Then the long-range feature clip with length L from the memory bank is formed as $L_t = \{l_{t-L+1}, ..., l_{t-1}, l_t\}$. In addition, the long-range feature clip is across the temporal variation block to refine the temporal and spatial information. The output is represented as $\bar{L}_t$. Procedure

The non-local operator is used to incorporate the temporal cues which contain the current feature c_t and long-range multi-scale feature $\bar{L}_t$. With the attention mechanism, the self-attention in the non-local block relates the long-range feature to the current frame feature. The architecture of non-local block is presented in Figure 4.4.

As depicted in Figure 4.4, the soft weights are determined for comparing the similarity between the current frame feature c_t and the refined feature $\bar{L}_t$. The non-local operation can be represented as Equation 4.1

$$y_t = Softmax(\frac{1}{Norm} f(c_t, \bar{L}_t)) g(\bar{L}_t)[17] \tag{4.1}$$

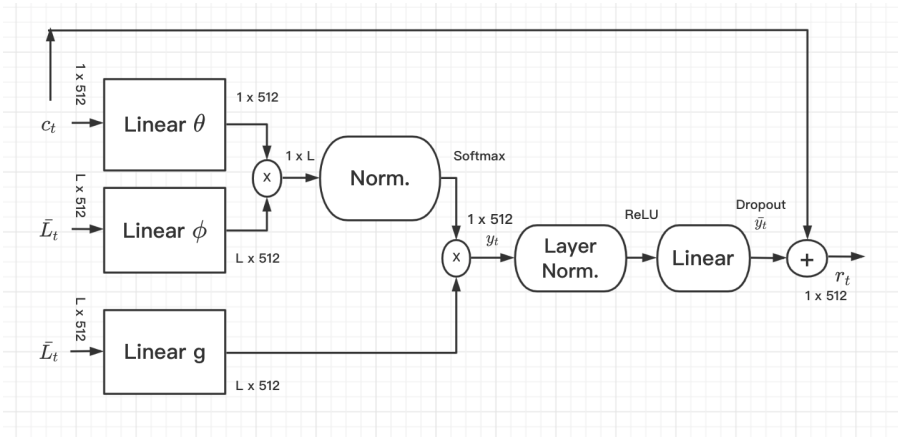


Figure 4.4: The architecture of non-local block

where $Norm$ is the normalization factor, and $f(c_t, \bar{L}_t) = e^{\theta {c_t}^T \phi(\bar{L}_t)}$, $\theta(c_t) = W_\theta c_t$, $\phi(\bar{L}_t) = W_\phi \bar{L}_t$, $g(\bar{L}_t) = W_g \bar{L}_t$.

The two linear embeddings similarity can be computed by an extension of Gaussian function, and the computation is conducted by matrix multiplication. In addition, the g operator is used for information transformation. This operation avoids heavy regularization by linearly projecting the input signal to a latent embedding space and then computing the similarity matrix directly on it. To enhance the regularization, the output of softmax function y_t will be refined by normalization and dropout, resulting in a new output $\bar{y}_t$. Through a connection, an enhanced output r_t is obtained by applying a non-local operator to connect the improved information and the current feature as depicted in Equation 4.2.

$$r_t = \bar{y}_t + c_t \tag{4.2}$$

After computing the enhanced representation r_t of long-range information, the final feature vector with 1024-dimension is obtained by concatenating the r_t with the current feature c_t. For final prediction, two fully connected layers are applied to the final part of the network. The current feature is important for the prediction results, and it is used in non-local operation and feature concatenation. Furthermore, the CataRCNet can predict the phase result with rich supportive information including the long-range temporal temporal dynamics and multi-scale temporal extent.

4.2 Loss Function

For multi-class classification, the cross entropy function is employed to calculate the loss. The training dataset includes 20 videos from "train06.mp4" to "train25.mp4". After down sampling to 1fps, the number of frames of each class (from step 0 to step 18) is 3845, 30, 95, 536, 365 ,862, 293, 896, 1359, 462 ,1304, 389 ,368, 247, 832, 785, 344, 170, and 978 respectively. For the unbalanced dataset, cross entropy loss function may not be ideal, since the error on minor

class might disappear with the domination of major class samples. To alleviate imbalance on the CATARACTS dataset, a Weighted Cross Entropy (WCE) function is chosen as the final loss function, presented in Equation 4.3.

$$WCE = -(\beta y log(\hat{y}) + (1-y) log(1-\hat{y})) \tag{4.3}$$

where β represents the adjustment weights.

By exploiting the adjustment weight, handicapping can be done either for false negatives or false positives. As per the number of each class, the weights of weighted cross entropy loss function for each class can be computed which are 0.0033, 0.4182, 0.1321, 0.0234, 0.0344, 0.0146, 0.0428, 0.0140, 0.0092, 0.0272, 0.0096, 0.0323, 0.0341, 0.0508, 0.0151, 0.0160, 0.0365, 0.0738, 0.0128.

4.3 Training and Testing

Training the entire model with a memory bank is not possible because of the complicated computations of back-propagating through the long-range memory bank. Therefore, the long-range memory bank is obtained by training Recurrent Convolutional Network (RCNet) based on SV-RCNet firstly. Here three backbone are chosen for recurrent convolutioanl network, including ResNet50, DenseNet169, ResNeSt50. For intialization, the weights of convolutional part are all initialized with a pretrained model on ImageNet, and sequential network is initialized with Xavier normal initializer. Then the long-range feature bank needs to be loaded to a pkl file, if it is the first time to train. Meanwhile, loading the information from memory bank will not disturb the end-to-end learning, but it needs more time for the first time. Consequently, the features of previous frames can be picked up from the bank without calculation during the training process. For inference, a sliding window is formed by creating the input frame clip sequentially, and it shifts a frame each time.

In the experiments, there are three types of networks for the phase recognition task: RCNet, RCNet+non-local operator, RCNet + Temporal variation layer (TVL) + non-local operator, and there are three backbone networks in each architecture including ResNet50, DenseNet169, and ResNeSt50. Therefore, nine models are trained for comparison.

Type 1 (RCNet): the baseline of the experiments is training RCNet with short-range input frames of 10 seconds:
(1) ResNet50 + LSTM; (2) DenseNet169 + LSTM; (3) ResNeSt + LSTM.

Type 2 (RCNet+ Non-local operator): it means to train the network with the supportive information of 30 seconds from long-range memory bank, and use the non-local operator to relate the long-range cues to the present feature:
(4) ResNet50 + LSTM + Non-local operator; (5) DenseNet169 + LSTM + Non-local operator; (6) ResNeSt + LSTM + Non-local operator.

Type 3 (RCNet + TVL + Non-local operator): This is the complete CataRCNet. The long-range and multi-scale information are employed, and the non-local operator is used to reference features:
(7) ResNet50 + LSTM + TVL+ Non-local operator; (8) DenseNet169 + LSTM + TVL + Non-local operator; (9) ResNeSt + LSTM + TVL + Non-local operator.

The framework is implemented based on Pytorch using a NVIDIA V100 GPU for training, and the CUDA version is 10.1. Firstly, the RCNet is trained with Adam optimizer and Weighted Cross Entropy loss function. The initial learning rate of RCNet is set to 5e-5 since the pre-trained backbone network are used in the training process. In general, the input clip lengths for RCNet and CataRCNet are both 10, and the batch sizes are both set to 100, due to the limitation of GPU Memory. For the first time to train CataRCNet, the long-range memory bank needs to be loaded to pkl files before training, but not disturbing the end-to-end learning. After obtaining the feature bank, we can access the temporal cues in feature bank from the pkl files. The length of supportive features L is set to 30 seconds since it can produce its maximal efficacy [17]. The optimizer and loss function of CataRCNet is the same as those of RCNet. For learning rate, the initial value is set to 1e-5.
The benchmark dataset CATARCTS is used in this research. The resolution of the input frames is 384×216 which is converted from the resolution of 1920×1080, and its frame rate is down sampled from about 30 fps to 1 fps.

For testing, 25 test videos in CATARACTS are utilized to evaluate the performance of developed networks. For comparison, the average results of 20 test videos are also summarized in Chapter 5 which are the same as those in benchmark challenge (MICCAI EndoVis Challenge 2020), including videos from "test01" to "test25" except "test01", "test07", "test14", "test16", "test19".

Chapter 5

Experimental Results and Discussions

5.1 Evaluation Metrics

For evaluation, the main metrics in surgical workflow recognition to evaluate the performance is f1-score, which is used in EndoVis Challenge 2020. In addition, accuracy, precision, and recall are also useful to quantitatively analyse the performance of the model.

The evaluation script for EndoVis / CATARACTS 2020 Challenge from [43] is utilized to evaluate the results. If this script is applied to this project, the ground truth and prediction results are required to be converted into the form of csv files, and they are in two directories "gt-step" and "pred-step". In the directories, the frame number and step information of 25 videos are divided into 25 csv files respectively, rather than the sum of all videos information. In this project, the frame rate is down sampled to 1 fps, so the frame number is "0, 30, 60..." in the csv file.

There are eighteen steps in cataract surgery, including step 1 to step 18, the names of which are 'Toric Marking', 'Implant Ejection', 'Incision', 'Viscodilatation', 'Capsulorhexis', 'Hydrodissetion', 'Nucleus Breaking', 'Phacoemulsification', 'Vitrectomy', 'Irrigation/Aspiration', 'Preparing Implant', 'Manual Aspiration', 'Implantation', 'Positioning', 'OVD Aspiration', 'Suturing', 'Sealing Control', and 'Wound Hydratation' respectively [43]. In addition, the step 0 which represents the "idle" is annotated in the ground truth file. However, the idle frames are not taken into account, so they are removed for evaluation part.

The f1-score, known as F-score or F-measure, is used to combine recall and precision, serving as a weighted average of the precision and recall [48]. The contributions of precision and recall to f1 score are equal. For all results, the value of 1 shows the best outcome, compared with 0, the worst one. In multi-class classification problem, the final value of f1 score is the average of the f1 score of each class, and "macro" is set to the average parameter in this project. The macro f1-score means to calculate the metrics for each class, and then calculate the unweighted mean.

The formula of F1 score is presented in Equation 5.1.

$$F1_score = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{5.1}$$

$$= \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{5.2}$$

where TP indicates the number of true positives, FP represents the number of false positives, TN stands for the number of true negatives and FN means the number of false negatives.

The formula of the recall is presented as Equation 5.3. In this formula, the ability of the classifier of finding all the positive samples is intuitively indicated.

$$Recall = \frac{TP}{TP + FN} \tag{5.3}$$

The precision formula could be described as Equation 5.4. Specifically, tp is the number of true positives, but fp demonstrates the false positives number. Precision is intuitively the ability of the classifier not to label negative samples as positive. The number of 1 and 0, shows the best and worst results respectively.

$$Precision = \frac{TP}{TP + FP} \tag{5.4}$$

For accuracy, it can be used as the metric of how well a classification test correctly identifies. The function of classification computes subset accuracy, where the set of labels for both special sample and y_true are required to be matched exactly. It is the same as the other three metrics, whose range of f1-score is from 0 to 1, where 1 indicates the best result, and 0 represents the worst one. In other words, the accuracy is the percentage of correct predictions within the total number of cases tested, and the formula for accuracy is shown in Equation 5.5.

$$Precision = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.5}$$

The f1 score is computed using "metrics.f1_score" based on the library "sklearn". Additionally, "metrics.recall_score" and metrics.precision_score from this library are also used to obtain the recall and precision values respectively.

5.2 Results and Discussion

Overall, nine models mentioned in Chapter 4.3 are trained using CATARACTS dataset. During the testing inference, the step predictions are output in the form of pkl files. Then the pkl files storing the predictions are used to produce the csv files which are required for the evaluation script.

The detailed test results of 25 videos using different methods are presented in Table A.1 - Table A.9 in the appendix, including the f1-score, accuracy, precision, and recall. In addition, the average f1 score of 25 test videos and that of 20 videos are both summarized at the bottom of each table. The performance of different architectures and backbones can be studied by the values of f1-score and accuracy. For evaluation, the average results of 25 test videos are concluded in Table 5.1.

Table 5.1: The results of 25 test videos for 9 models

no.	Model	F1-score	Accuracy	Precision	Recall
1	ResNet+LSTM	**0.7796**	0.8236	0.9149	0.8814
2	DenseNet+LSTM	0.7570	0.8135	0.9073	0.8734
3	ResNeSt+LSTM	0.7641	0.8042	0.9206	0.8692
4	ResNet+LSTM + NL	0.7779	0.8217	0.9269	0.8767
5	DenseNet+LSTM + NL	0.7604	0.8190	0.9112	0.8744
6	ResNeSt+LSTM + NL	0.7399	0.7818	0.9194	0.8466
7	ResNet+LSTM +TVL + NL	0.7384	0.7867	0.9108	0.8555
8	DenseNet+LSTM +TVL+ NL	0.7590	**0.8253**	0.9146	0.8714
9	ResNeSt+LSTM +TVL+ NL	0.7613	0.8074	0.9226	0.8624

As presented in Table 5.1, F1-score for M1 is 0.7796, the highest of the nine models, and M7 exhibits the lowest f1-score of 0.7384. In addition, M8 is the one with the highest accuracy which is 0.8253 out of the nine models. In the analysis of the 25 test results of 9 models, the arrangement of this should be chiefly demonstrated. Horizontally, the 9 models can be divided into 3 groups: M1, M2, M3 as the first group, M4, M5, M6 as the second group, and M7, M8, M9 as the third group. The reason for this grouping is that each group shares the same architecture with different backbones. Therefore, the horizontal comparison is within the group to figure out the differences of results by using distinguished backbones. Unlike the above, the vertical analysis is the comparison across groups as it aims to evaluate the values of the same backbones result within different architectures. For example, the vertical comparison could choose model 1, 4, 7 as a comparative reference group because they utilize ResNet as their backbone but with different architectures.

In order to evaluate intuitively, the six groups mentioned above created to compare the results are plotted in six bar charts. They demonstrate the values of F1-score that correspond to 25 test videos shown in Figure 5.1 - Figure 5.6.

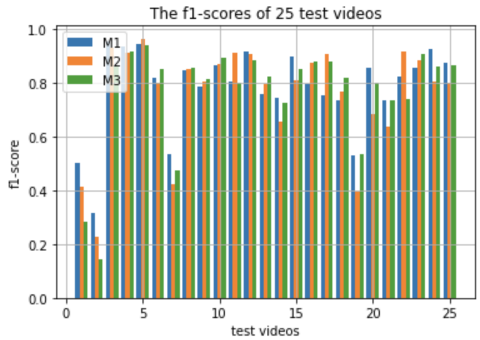


Figure 5.1: F1-scores of 25 test videos of M1, M2, M3

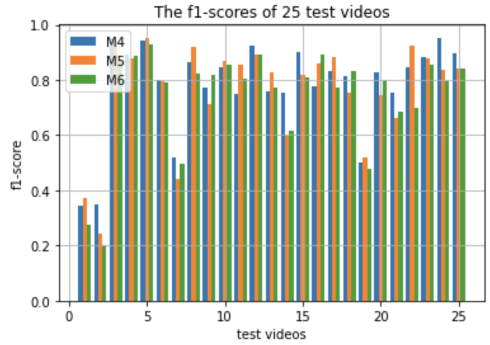


Figure 5.2: F1-scores of 25 test videos of M4, M5, M6

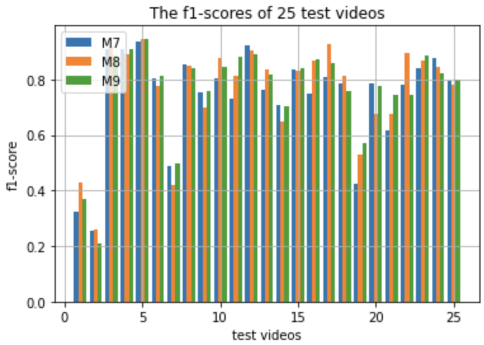


Figure 5.3: F1-scores of 25 test videos of M7, M8, M9

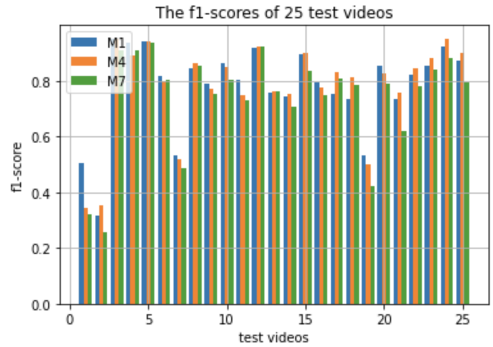


Figure 5.4: F1-scores of 25 test videos of M1, M4, M7

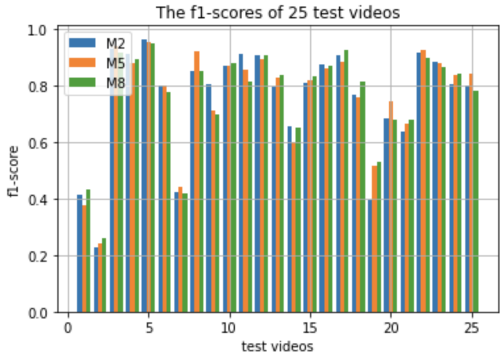


Figure 5.5: F1-scores of 25 test videos of M2, M5, M8

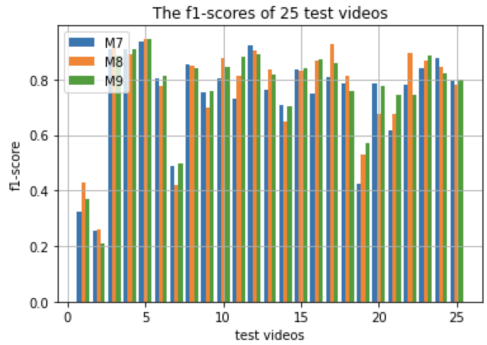


Figure 5.6: F1-scores of 25 test videos of M3, M6, M9

Having clearly introduced the arrangement, the results obtained could provide instruction in terms of their effectiveness and accuracy. Turning to the values of F1-score, it can be obvious noted that the highest value is 0.7796, which is in group 1, meaning that the performance of ResNet50 +

LSTM model is the best in this experiment. By calculating the average of each group, the highest number is as well in group 1, 0.7669, compared with the 0.7594 in group 2, and 0.7529 in group 3. Therefore, the architecture of group 1 may probably be the most suitable. It indicates that RCNet performs best for cataract surgery in this experiment. It also means that the long-range memory bank and multi-scale kernels do not improve the results as expected.

From the perspective of vertical view, the grouping is mainly focused on the effectiveness of ResNet, DenseNet and ResNeSt, representing Model 1,4,7, Model 2,5,8 and Model 3,6,9 respectively. According to the average value of each group, the ResNet achieved the score of 0.7653, higher than 0.7520 of DenseNet and 0.7551 of ResNeSt. This means that ResNet is the most proper backbone in the experiment. The average values of these three groups are very close, meaning that the more powerful convolutional networks (DenseNet and ResNeSt) do not improve the classification performance significantly.

When analyzing the accuracy part, the method here is similar with the above F1-score analysis, thereby a simple result given. Although model 8 have the most outstanding data, which is 0.8253, the group of which in horizontal analysis experiences the lowest average number of 0.8064, compared with the 0.8138 in group 1, and 0.8075 in group 2. In terms of vertical analysis, the DenseNet group shows that DenseNet has a strong connect with the accuracy value as it achieves the 0.8193, higher than 0.8107 of ResNet group and 0.7978 of ResNeSt group.

As shown in Figure 5.1 - Figure 5.6, there is a common and notable phenomenon that in video 1, 2, 7 and 19, all the values are below 0.6, which means that a challenge exists in the algorithm to process these videos. To further explore the reason, the detailed information of test dataset is recorded in Table 5.2.

Table 5.2: Cataracts: the ground truth files of test dataset

Ground truth files	Number of frames	With Steps	With Special Steps
Test01	22620	0,1, 3-8, 10, 11, 13-15, 18	1, 11
Test02	55711	0, 3-8, 7,10, 11,12,13-15, 16,17,18	9,11,12,16,17
Test03	15093	0, 3-8,10,11, 13-15, 18	11
Test04	14513	0, 3-8,10, 13-15, 18	-
Test05	14074	0, 3-8,10, 11,13-15, 18	11
Test06	16978	0, 3-8,10, 11,13-15, 17,18	11,17
Test07	31203	0, 3-8,9-12, 13,14,16, 18	9,11,12,16
Test08	13298	0, 3-8,10, 13-15, 17,18	17
Test09	19362	0, 3-8,10, 11 13,14, 18	11
Test10	17055	0, 3-8,10, 11,13-15, 18	11
Test11	15420	0, 3-8,10, 11,13-15, 18	11
Test12	11741	0, 3-8,10, 11,13-15, 18	11
Test13	16683	0,1, 3-8,10, 14,15, 18	1
Test14	11827	0, 3-8,10, 13-15,17, 18	17
Test15	14531	0, 3-8,10, 11,13-15, 18	11
Test16	20470	0, 3-8,10,11, 13-15, 16,18	11 ,16
Test17	15353	0,2, 3-8,10, 13-15, 16,18	2,16
Test18	20973	0, 3-8,10, 13-15, 18	-
Test19	21979	0, 2,3-8,10,11, 13-16, 18	2,11,16
Test20	16028	0, 3-8,10, 11,13-15, 18	11
Test21	14416	0, 3-8,10,11, 13-15, 18	11
Test22	14813	0, 3-8,10, 13-15, 16,18	16
Test23	15669	0, 3-8,10, 13-15, 18	-
Test24	15490	0, 3-8,10, 11,13-15, 18	11
Test25	17716	0, 3-8,10,11, 13-15, 18	11

As Table 5.2 depicted, test video 1, video 2, video 7, and video 19 are complicated, as these videos all have special steps and more frames. Test 01 has special steps with 1 and 11, and Test 02 has special steps with 9, 11, 12, 16, and 17. In addition, the special steps of Test 07 are 9, 11, 12, 16, and those of Test 19 are 2, 11, 16. According to Table 3.2 in Chapter 3.2, only video train 19 has step 9 and step 12 in training set, and only train 24 has step 1. This table also illustrates that a few videos which have step 2, 16, 17 are in the training set. Therefore, it can be explained that the insufficient training data of these steps lead to the unsatisfied test results of the test videos which have these steps.

In order to ensure the feasibility and adaption of the algorithm, it is referenceable to compare with the result of a previous game named EndoVis2020 Challenge, shown in Table 5.3. In this competition, the average f1-scores of all members range from 0.892 of ARTC team to 0.7827 of Uniandes_BCV team. As a reference, the results of 9 test models are somewhere within the range above, especially for model 4, whose result is up to 0.8275, exceeding that of 0.8242 achieved by

the second place, CAMI SIAT team.

Table 5.3: The results of 20 test videos for 9 models [30]

Model	M1	M2	M3	M4	M5
f1-score	0.81924	0.808215	0.81066	**0.8275**	0.8111
Model	M6	M7	M8	M9	-
f1-score	0.7850	0.7885	0.80397	0.8014	-
Team	ARTG	CAMI SIAT	SK	Uniandes_BCV	-
f1-score	**0.892**	0.8242	0.8181	0.7827	-

Table 5.4 demonstrates the process of adaptation of the loss function. Before adjustment, the use of Cross Entropy loss function caused an unpleasant result with the value of 0.6942 in Model 1 F1-score and 0.6935 in Model 4 F1-score. The reason for this is primarily due to the limited frame number in step one so that there does not exist enough training for the network, leading to the poor performance result. To adjust, the Weighted Cross Entropy has been applied and the situation gets much better, with the values turning to 0.7796 and 0.7779 in Model 1 and 4 respectively.

Table 5.4: The results of 25 test videos using different loss function for M1 and M4

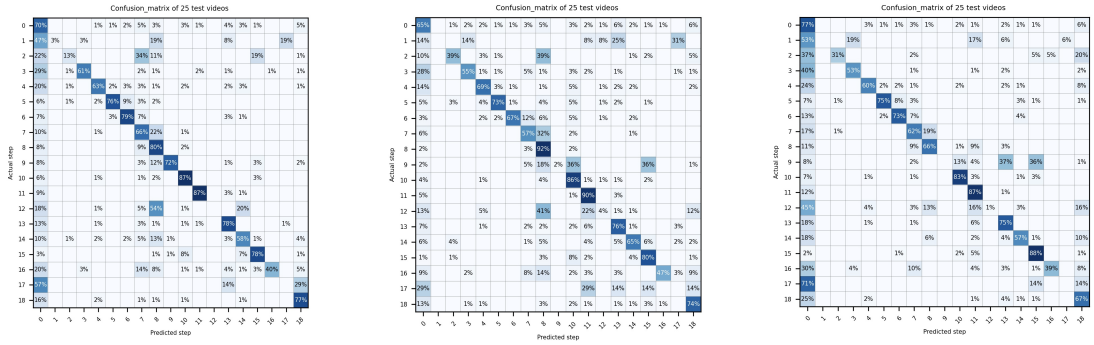
Loss function	Weighted Cross Entropy	Cross Entropy
f1-score (M1)	0.7796	0.6942
f1-score (M4)	0.7779	0.6935

Table 5.5 describes the test results of ResNet50 + LSTM via different input lengths. As can be seen from the table, the increase of input clip length from 10 to 30, negatively influences the F1-scores, resulting in the decrease from 0.7796 to 0.7129. The interesting thing also happened in the accuracy part. With the input clip length increasing, the value of accuracy experiences an increase from 0.8236 to 0.8308, but immediately dropping to 0.8066, from the peak of 20 input clip length to 30 input clip length. A probable reason for this case is that long range temporal information might have more noise, leading to no improvement with long input clip length.

Table 5.5: The results of 25 test videos of ResNet50 + LSTM

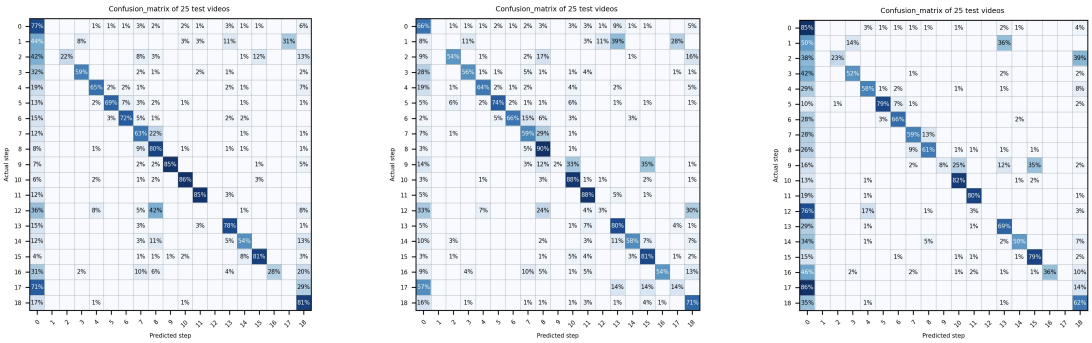
Input clip Length (s)	F1-score	Accuracy	Precision	Recall
10	**0.7796**	0.8236	0.9149	0.8814
20	0.7748	**0.8308**	0.9157	0.8797
30	0.7129	0.8066	0.9022	0.8433

There are 9 charts, showing in Figure 5.7 - 5.9 to describe the confusion matrix analysis, corresponding to 9 models repetitively. For explanation, the confusion matrix is a tool for better understanding of what classification algorithm has a good performance while what suffers from some errors. And the inference can be illustrated from the logic that by combining the ordinate, which stands for the ground truth in each step, and the abscissa, which represents the predicted value of the model in each step, their proportion of overlap will tell how accurate the model is. Therefore, in this matrix, the values on the diagonal are required to be paid attention on for comparison. And the nine charts are presented in a 3 by 3 arrangement, in order to analyze the horizontal and vertical aspects as the horizontal figures share the same architecture while the vertical ones share the same backbones. Firstly, in a general view of the nine models, there exist apparent blank spaces or nearly blank spaces in the steps like 1, 12 and 17. This is because in these steps, the number of frames in the training dataset is severely limited and lacked, making all the models not experience the deep and full training process, thereby the unexpected performance. The second feature can be concluded from the vertical view in Step 9, which is the comparison of ResNet, DenseNet and ResNeSt. There is a sharp contrast that the ResNet has an excellent performance, above 70% of accuracy, while the values of others are far away from that, approximately below 10%. One possible reason for this phenomenon may probably be that the weighted loss function is better matched with ResNet and it has a positive impact on the outcome, whereas it does not improve the results of the other two models significantly. Next, the third discovery is in terms of the relative better outcome of DenseNet in step 2, 12 and 16, in contrast to others. Especially for step 12, though the results of DenseNet are around 5%, it is much better than other models, whose results are almost 0%. Another finding shows that in model 1, the accuracy value of step 2 is 13%, but the value of 34%, computed by matching actual step 2 and predicted step 7, is much higher. This may because some similarities are shared in the two steps: Implant Ejection and Nucleus Breaking, as these two processes are using the same tools and surgical actions, causing a hard distinction between them during the training. The same reason could also be applied into the step 12 and 8, which means the algorithm needs to be improved.



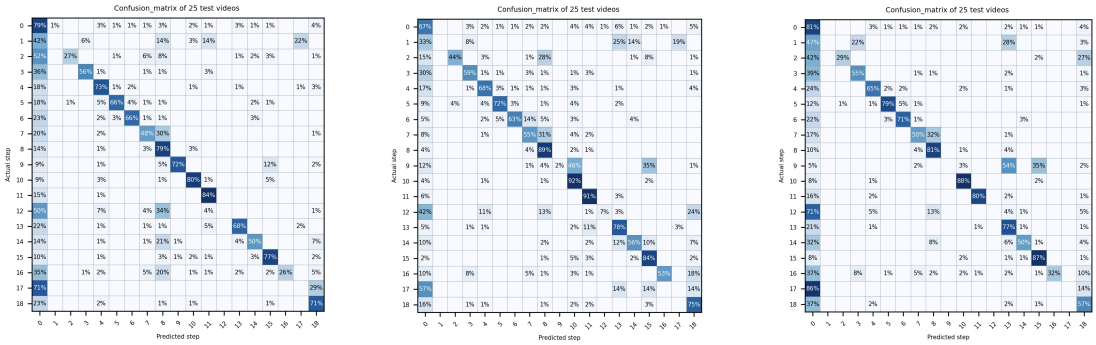
(a) M1 (b) M2 (c) M3

Figure 5.7: Confusion matrix of M1, M2, M3



(a) M4 (b) M5 (c) M6

Figure 5.8: Confusion matrix of M4, M5, M6



(a) M7 (b) M8 (c) M9

Figure 5.9: Confusion matrix of M7, M8, M9

In order to further explore results intuitively, the prediction results produced by Model 4 during four complete cataract surgery videos including video 1, 16, 10, and 24 are shown in Figure 5.10 - Figure 5.13. For Model 4, video 1 has the lowest f1-score 0.3433, and video 24 has the highest f1-score 0.9516. And the f1-scores of video 16 and video 20 are 0.7775 and 0.8280 respectively. It can be found that video 24 has smoother results with more consistent step predictions than other videos. Figure 5.10 illustrates that many steps are wrongly predicted to step 14, step 18.

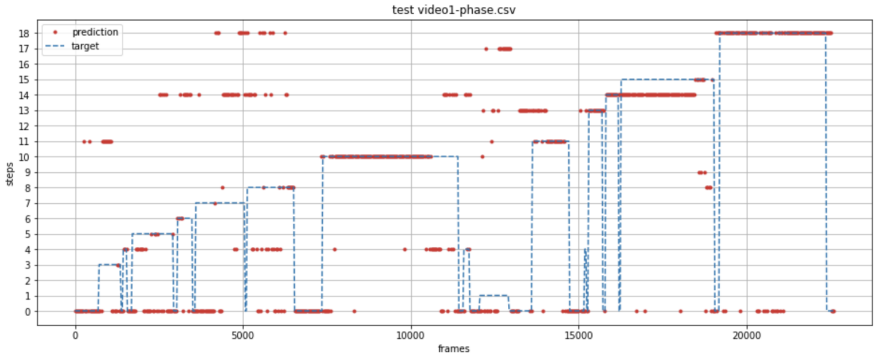


Figure 5.10: Prediction results during test video 1 for M4 with f1-score 0.3433

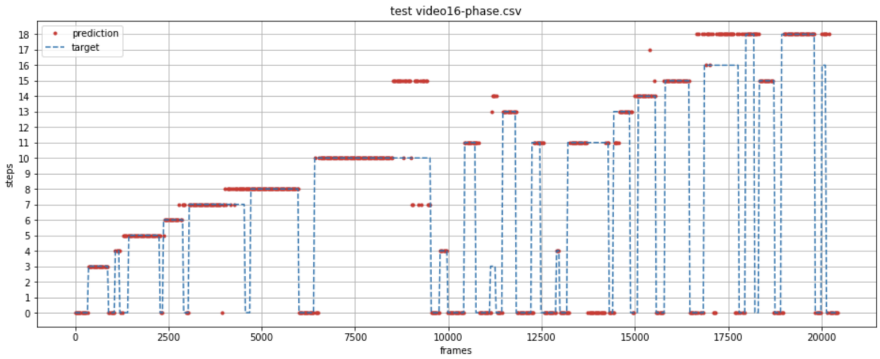


Figure 5.11: Prediction results during test video 16 for M4 with f1-score 0.7775

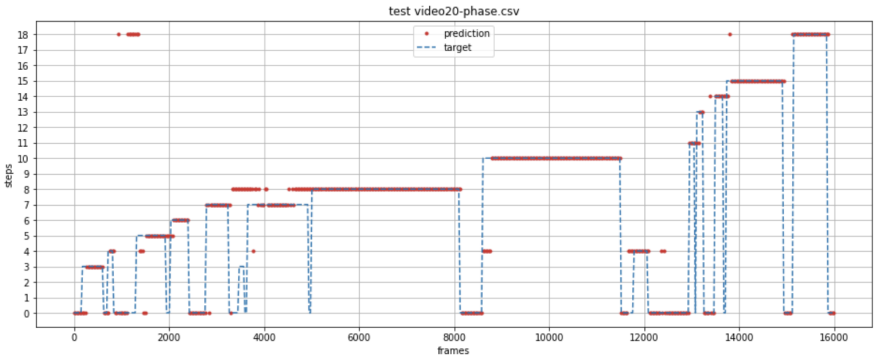


Figure 5.12: Prediction results during test video 20 for M4 with f1-score 0.8280

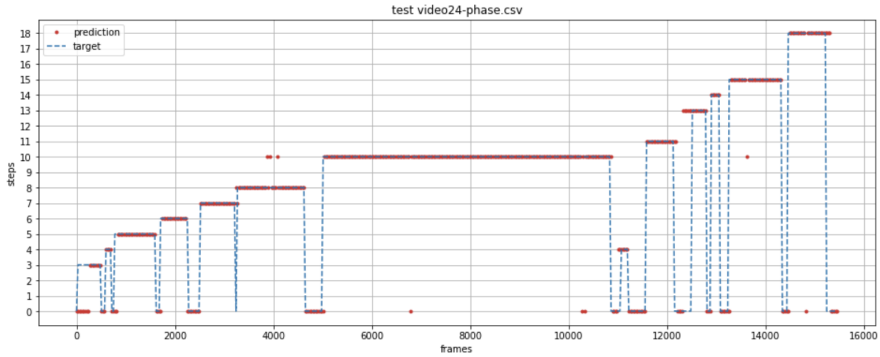


Figure 5.13: Prediction results during test video 24 for M4 with f1-score 0.9516

Chapter 6

Conclusions and Future Work

In conclusion, the end-to-end recurrent convolutional networks are developed for cataract surgical workflow analysis based on a temporal memory relation network. The aim of this project is to leverage supervised deep learning algorithms to classify cataracts video frames according to their surgical phases, referred to as surgical phase recognition. In this project, nine methods are designed to recognize the steps of cataract surgery, and they all can achieve the experimental goal. To be specific, three different network architectures (RCNet, RCNet + Non-local operator, RCNet + Non-local operator + temporal variant layer) with three backbones (ResNet50, DenseNet169, ResNeSt50) are developed for classification task. The non-local operator and temporal variant layer are designed to provide long-range and multi-scale temporal supportive information for accurate classification without disturbing the end-to-end learning process.

Compared with the results of EndoVis 2020 Challenge, the average f1-scores of 9 models for 20 test videos are within the range of the Challenge results. The best results among the nine methods are produced by M4 (ResNet + LSTM + non-local operator), whose value is up to 0.8275, exceeding that of 0.8242 achieved by the second place, CAMI SIAT team in Challenge. Therefore, the results of the nine models are all acceptable and reasonable.

The contributions of this research are:
(1) For the first time, the temporal memory relation network is applied to cataract surgery, referred to as CataRCNet.
(2) For the first time, the DenseNet169 is selected as the backbone for the end-to-end recurrent convolutional network, referred to as SV-DenseRCNet.
(3) It is proved that the nine developed methods in this project can be used for cataract surgical workflow analysis.

For 25 test videos, the highest f1 score is 0.7796 produced by M1 (ResNet + LSTM), meaning that the end-to-end RCNet network is a promising architecture in the surgical workflow analysis task. However, the non-local operator and temporal variant layer do not improve the classification results as expected, meaning that the temporal supportive information does not help the network to recognize more accurately in the experiments. The reason for this phenomenon might be that

long-term information has more noisy information which has a negative impact on the recognition results. In addition, keeping increasing the input clip length cannot always increase the f1-scores. Besides, the networks with ResNet50 backbone can recognize step 9 after using weighted cross entropy loss function, but the networks with DenseNet or ResNeSt still cannot recognize step 9. Therefore, the performance of ResNet is better than others in this experiment. However, the DenseNet has better recognition results than other backbones in step 2, 12, 16, indicating that the networks with DenseNet as backbone have the potential to produce better results if they can solve the issue of low accuracy in step 9.

In the future, it is planned to improve the predicted results by post-processing strategies, such as prior knowledge inference in [25]. Expect from some a number of resting frames and frames with slight motions, there are some frames with various artifacts in the video, and they are always hard to be judged. Through analysis, it is summarized that these frames are ordered in a sequential manner, which can be predicted to some degree. Because that the change of key action brings much information in terms of time, the next frame can probably be predicted via prior knowledge inference (PKI). By comparing the predicted results with the actual ones, the final results can be accurately corrected. Furthermore, the sequential model Long-Short Term Memory (LSTM) can be replaced by a more powerful model, Transformer, to improve the performance.

Bibliography

[1] S. Robertson, "Cataract epidemiology," 2019. [Online]. Available: https://www.news-medical.net/health/Cataract-Epidemiology.aspx

[2] M. Wormstone, "Strategies to improve cataract surgery outcomes," 2019. [Online]. Available: https://www.news-medical.net/health/Strategies-to-Improve-Cataract-Surgery-Outcomes.aspx

[3] S. Trikha, A. Turnbull, R. Morris, D. Anderson, and P. Hossain, "The journey to femtosecond laser-assisted cataract surgery: new beginnings or a false dawn?" *Eye*, vol. 27, no. 4, pp. 461–473, 2013.

[4] H. Al Hajj, M. Lamard, P.-H. Conze, S. Roychowdhury, X. Hu, G. Maršalkaitė, O. Zisimopoulos, M. A. Dedmari, F. Zhao, J. Prellberg *et al.*, "Cataracts: Challenge on automatic tool annotation for cataract surgery," *Medical image analysis*, vol. 52, pp. 24–41, 2019.

[5] N. Padoy, "Machine and deep learning for workflow recognition during surgery," *Minimally Invasive Therapy & Allied Technologies*, vol. 28, no. 2, pp. 82–90, 2019.

[6] L. Maier-Hein, S. S. Vedula, S. Speidel, N. Navab, R. Kikinis, A. Park, M. Eisenmann, H. Feussner, G. Forestier, S. Giannarou *et al.*, "Surgical data science for next-generation interventions," *Nature Biomedical Engineering*, vol. 1, no. 9, pp. 691–696, 2017.

[7] A. Huaulmé, P. Jannin, F. Reche, J.-L. Faucheron, A. Moreau-Gaudry, and S. Voros, "Offline identification of surgical deviations in laparoscopic rectopexy," *Artificial Intelligence in Medicine*, vol. 104, p. 101837, 2020.

[8] A. P. Twinanda, S. Shehata, D. Mutter, J. Marescaux, M. De Mathelin, and N. Padoy, "Endonet: a deep architecture for recognition tasks on laparoscopic videos," *IEEE transactions on medical imaging*, vol. 36, no. 1, pp. 86–97, 2016.

[9] S. Kumar, P. Singhal, and V. N. Krovi, "Computer-vision-based decision support in surgical robotics," *IEEE Design & Test*, vol. 32, no. 5, pp. 89–97, 2015.

[10] A. J. Hung, J. Chen, and I. S. Gill, "Automated performance metrics and machine learning algorithms to measure surgeon performance and anticipate clinical outcomes in robotic surgery," *JAMA surgery*, vol. 153, no. 8, pp. 770–771, 2018.

[11] S. Panesar, Y. Cagle, D. Chander, J. Morey, J. Fernandez-Miranda, and M. Kliot, "Artificial intelligence and the future of surgical robotics," *Annals of surgery*, vol. 270, no. 2, pp. 223–226, 2019.

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[13] S. P. Devi, K. S. Rao, and S. S. Sangeetha, "Prediction of surgery times and scheduling of operation theaters in optholmology department," *Journal of medical systems*, vol. 36, no. 2, pp. 415–430, 2012.

[14] K. Schoeffmann, M. Taschwer, S. Sarny, B. Münzer, M. J. Primus, and D. Putzgruber, "Cataract-101: video dataset of 101 cataract surgeries," in *Proceedings of the 9th ACM multimedia systems conference*, 2018, pp. 421–425.

[15] A. Achiron, F. Haddad, M. Gerra, E. Bartov, and Z. Burgansky-Eliash, "Predicting cataract surgery time based on preoperative risk assessment," *European journal of ophthalmology*, vol. 26, no. 3, pp. 226–229, 2016.

[16] M. Lanza, R. Koprowski, R. Boccia, K. Krysik, S. Sbordone, A. Tartaglione, A. Ruggiero, and F. Simonelli, "Application of artificial intelligence in the analysis of features affecting cataract surgery complications in a teaching hospital," *Frontiers in Medicine*, vol. 7, 2020.

[17] Y. Jin, Y. Long, C. Chen, Z. Zhao, Q. Dou, and P.-A. Heng, "Temporal memory relation network for workflow recognition from surgical video," *IEEE Transactions on Medical Imaging*, 2021.

[18] F. Lalys, L. Riffaud, D. Bouget, and P. Jannin, "A framework for the recognition of high-level surgical tasks from video images for cataract surgeries," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 4, pp. 966–976, 2011.

[19] T. Czempiel, M. Paschali, M. Keicher, W. Simson, H. Feussner, S. T. Kim, and N. Navab, "Tecno: Surgical phase recognition with multi-stage temporal convolutional networks," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2020, pp. 343–352.

[20] N. Padoy, T. Blum, S.-A. Ahmadi, H. Feussner, M.-O. Berger, and N. Navab, "Statistical modeling and recognition of surgical workflow," *Medical image analysis*, vol. 16, no. 3, pp. 632–641, 2012.

[21] C. Meißner, J. Meixensberger, A. Pretschner, and T. Neumuth, "Sensor-based surgical activity recognition in unconstrained environments," *Minimally Invasive Therapy & Allied Technologies*, vol. 23, no. 4, pp. 198–205, 2014.

[22] O. Zisimopoulos, E. Flouty, I. Luengo, P. Giataganas, J. Nehme, A. Chow, and D. Stoyanov, "Deepphase: surgical phase recognition in cataracts videos," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2018, pp. 265–272.

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] G. Yengera, D. Mutter, J. Marescaux, and N. Padoy, "Less is more: Surgical phase recognition with less annotations through self-supervised pre-training of cnn-lstm networks," *arXiv preprint arXiv:1805.08569*, 2018.

[25] Y. Jin, Q. Dou, H. Chen, L. Yu, J. Qin, C.-W. Fu, and P.-A. Heng, "Sv-rcnet: workflow recognition from surgical videos using recurrent convolutional network," *IEEE transactions on medical imaging*, vol. 37, no. 5, pp. 1114–1126, 2017.

[26] A. P. Twinanda, "Vision-based approaches for surgical activity recognition using laparoscopic and rbgd videos," Ph.D. dissertation, Strasbourg, 2017.

[27] C. Lea, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks: A unified approach to action segmentation," in *European Conference on Computer Vision*. Springer, 2016, pp. 47–54.

[28] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[29] Y. A. Farha and J. Gall, "Ms-tcn: Multi-stage temporal convolutional network for action segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3575–3584.

[30] Presentation, "Cataracts2020 endoscopic vision challenge," 2020. [Online]. Available: https://cataracts2020.grand-challenge.org

[31] Y. Jin, H. Li, Q. Dou, H. Chen, J. Qin, C.-W. Fu, and P.-A. Heng, "Multi-task recurrent convolutional network with correlation loss for surgical video analysis," *Medical image analysis*, vol. 59, p. 101572, 2020.

[32] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6202–6211.

[33] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Technology (ICET)*, 2017, pp. 1–6.

[34] S. Haykin and N. Network, "A comprehensive foundation," *Neural networks*, vol. 2, no. 2004, p. 41, 2004.

[35] E. E. J. Lepe and A. M. Vázquez, "A beginner's tutorial for cnn," 2017.

[36] Y. Bengio, I. Goodfellow, and A. Courville, *Deep learning*. MIT press Massachusetts, USA:, 2017, vol. 1.

[37] J. Wu, "Introduction to convolutional neural networks," *National Key Lab for Novel Software Technology. Nanjing University. China*, vol. 5, no. 23, p. 495, 2017.

[38] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[39] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[40] S. Hochreiter and J. Schmidhuber, "Lstm can solve hard long time lag problems," *Advances in neural information processing systems*, pp. 473–479, 1997.

[41] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with lstm recurrent networks," *Journal of machine learning research*, vol. 3, no. Aug, pp. 115–143, 2002.

[42] R. C. Staudemeyer and E. R. Morris, "Understanding lstm–a tutorial into long short-term memory recurrent neural networks," *arXiv preprint arXiv:1909.09586*, 2019.

[43] H. ALHAJJ, M. Lamard, P.-h. Conze, B. Cochener, and G. Quellec, "Cataracts," 2021. [Online]. Available: https://dx.doi.org/10.21227/ac97-8m18

[44] K. Charrière, G. Quellec, M. Lamard, D. Martiano, G. Cazuguel, G. Coatrieux, and B. Cochener, "Real-time analysis of cataract surgery videos using statistical models," *Multimedia Tools and Applications*, vol. 76, no. 21, pp. 22 473–22 491, 2017.

[45] H. Zhang, C. Wu, Z. Zhang, Y. Zhu, H. Lin, Z. Zhang, Y. Sun, T. He, J. Mueller, R. Manmatha *et al.*, "Resnest: Split-attention networks," *arXiv preprint arXiv:2004.08955*, 2020.

[46] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[47] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7794–7803.

[48] C. Goutte and E. Gaussier, "A probabilistic interpretation of precision, recall and f-score, with implication for evaluation," in *European conference on information retrieval*. Springer, 2005, pp. 345–359.

Appendix A

The results of 9 models

A.1 ResNet50 + LSTM

Table A.1: Test results of ResNet50 + LSTM

Videos	F1-score	Accuracy	Precision	Recall
Test01	0.5034	0.6088	0.8666	0.7618
Test02	0.3148	0.4841	0.7012	0.6283
Test03	0.9514	0.9622	0.9962	0.9584
Test04	0.9350	0.9317	0.9800	0.9546
Test05	0.9432	0.9510	0.9808	0.9631
Test06	0.8172	0.9009	0.9224	0.9029
Test07	0.5321	0.6058	0.8057	0.7606
Test08	0.8461	0.9233	0.9305	0.9198
Test09	0.7873	0.8031	0.9066	0.9069
Test10	0.8649	0.8310	0.9531	0.9246
Test11	0.8036	0.8638	0.9020	0.9029
Test12	0.9170	0.9284	0.9824	0.9397
Test13	0.7568	0.7572	0.8904	0.8737
Test14	0.7454	0.8934	0.9092	0.8639
Test15	0.8971	0.8760	0.9672	0.9348
Test16	0.7937	0.8226	0.9285	0.8887
Test17	0.7537	0.8004	0.9161	0.8593
Test18	0.7356	0.7825	0.8786	0.8598
Test19	0.5306	0.6352	0.7725	0.7629
Test20	0.8542	0.8876	0.9286	0.9335
Test21	0.7341	0.7937	0.9114	0.8632
Test22	0.8221	0.8398	0.9587	0.8972
Test23	0.8551	0.8908	0.9499	0.9112
Test24	0.9250	0.9244	0.9862	0.9431
Test25	0.8716	0.8932	0.9470	0.9210
Average of 25 videos	0.7796	0.8236	0.9149	0.8814
Average of 20 videos	0.81924	-	-	-

A.2 DenseNet169 + LSTM

Table A.2: Test results of DenseNet169 + LSTM

Videos	F1-score	Accuracy	Precision	Recall
Test01	0.4148	0.5836	0.7745	0.7364
Test02	0.2259	0.4276	0.6610	0.6081
Test03	0.9292	0.9563	0.9663	0.9647
Test04	0.9110	0.9089	0.9746	0.9431
Test05	0.9631	0.9638	0.9841	0.9783
Test06	0.8002	0.9115	0.9109	0.8929
Test07	0.4235	0.5673	0.7441	0.7247
Test08	0.8503	0.9255	0.9276	0.9236
Test09	0.8029	0.8202	0.9134	0.9006
Test10	0.8685	0.8644	0.9566	0.9369
Test11	0.9113	0.9183	0.9732	0.9418
Test12	0.9071	0.9361	0.9885	0.9305
Test13	0.7940	0.7590	0.9049	0.8978
Test14	0.6543	0.8122	0.8958	0.8189
Test15	0.8105	0.8120	0.9417	0.9048
Test16	0.8737	0.8959	0.9515	0.9267
Test17	0.9042	0.9237	0.9638	0.9437
Test18	0.7651	0.8612	0.9056	0.8709
Test19	0.3944	0.4522	0.7701	0.6805
Test20	0.6848	0.8127	0.8672	0.8547
Test21	0.6358	0.7167	0.9078	0.8055
Test22	0.9130	0.9168	0.9851	0.9352
Test23	0.8846	0.8793	0.9480	0.9315
Test24	0.8058	0.8857	0.9378	0.9007
Test25	0.7971	0.8271	0.9272	0.8822
Average of 25 videos	0.7570	0.8135	0.9073	0.8734
Average of 20 videos	0.808215	-	-	-

A.3 ResNeSt50 + LSTM

Table A.3: Test results of ResNeSt50 + LSTM

Videos	F1-score	Accuracy	Precision	Recall
Test01	0.2823	0.4682	0.7230	0.6522
Test02	0.1415	0.3640	0.6548	0.5668
Test03	0.8613	0.8986	0.9671	0.9061
Test04	0.9173	0.9027	0.9750	0.9461
Test05	0.9390	0.9211	0.9772	0.9597
Test06	0.8481	0.9097	0.9368	0.9091
Test07	0.4725	0.5548	0.7875	0.7415
Test08	0.8555	0.9345	0.9396	0.9194
Test09	0.8134	0.8341	0.9371	0.9077
Test10	0.8924	0.8398	0.9875	0.9216
Test11	0.8006	0.8405	0.9634	0.8660
Test12	0.8831	0.9233	0.9766	0.9234
Test13	0.8235	0.9083	0.9312	0.8972
Test14	0.7241	0.8299	0.9045	0.8470
Test15	0.8483	0.8202	0.9575	0.9055
Test16	0.8785	0.8856	0.9587	0.9254
Test17	0.8766	0.8865	0.9732	0.9119
Test18	0.8153	0.8641	0.9565	0.8914
Test19	0.5319	0.5219	0.8255	0.7629
Test20	0.8001	0.8258	0.9349	0.8979
Test21	0.7333	0.7500	0.9184	0.8587
Test22	0.7404	0.8235	0.9281	0.8517
Test23	0.9048	0.9080	0.9786	0.9337
Test24	0.8576	0.8702	0.9695	0.9132
Test25	0.8620	0.8186	0.9538	0.9130
Average of 25 videos	0.7641	0.8042	0.9206	0.8692
Average of 20 videos	0.81066	-	-	-

A.4 ResNet50+ LSTM + Non-local operator

Table A.4: Test results of ResNet50 + LSTM+ Non-local

Videos	F1-score	Accuracy	Precision	Recall
Test01	0.3433	0.5186	0.8157	0.6938
Test02	0.3511	0.5148	0.7515	0.6451
Test03	0.9497	0.9563	0.9952	0.9572
Test04	0.8904	0.8923	0.9797	0.9204
Test05	0.9432	0.9510	0.9775	0.9644
Test06	0.7978	0.9027	0.9256	0.8886
Test07	0.5205	0.6260	0.8330	0.7495
Test08	0.8634	0.9391	0.9450	0.9180
Test09	0.7722	0.7984	0.9030	0.9024
Test10	0.8470	0.8063	0.9585	0.9060
Test11	0.7484	0.8346	0.9473	0.8534
Test12	0.9230	0.9386	0.9899	0.9395
Test13	0.7607	0.7734	0.8933	0.8768
Test14	0.7549	0.8756	0.9171	0.8564
Test15	0.8991	0.8781	0.9807	0.9309
Test16	0.7775	0.8123	0.9247	0.8892
Test17	0.8322	0.8571	0.9488	0.8928
Test18	0.8138	0.8069	0.9355	0.8877
Test19	0.5007	0.5738	0.7981	0.7465
Test20	0.8282	0.8783	0.9261	0.9176
Test21	0.7553	0.8021	0.9266	0.8641
Test22	0.8443	0.8621	0.9816	0.9021
Test23	0.8819	0.8831	0.9576	0.9243
Test24	0.9516	0.9535	0.9953	0.9612
Test25	0.8983	0.9068	0.9666	0.9302
Average of 25 videos	0.7779	0.8217	0.9269	0.8767
Average of 20 videos	0.8275	-	-	-

A.5 DenseNet169 + LSTM +Non-local operator

Table A.5: Test results of DenseNet169 + LSTM +Non-local operator

Videos	F1-score	Accuracy	Precision	Recall
Test01	0.3738	0.5411	0.7315	0.7186
Test02	0.2416	0.4663	0.7100	0.6170
Test03	0.9406	0.9523	0.9824	0.9586
Test04	0.8771	0.8716	0.9674	0.9271
Test05	0.9531	0.9595	0.9856	0.9681
Test06	0.7976	0.8991	0.9192	0.9018
Test07	0.4408	0.5856	0.7529	0.7268
Test08	0.9179	0.9413	0.9641	0.9641
Test09	0.7125	0.7953	0.8773	0.8557
Test10	0.8678	0.8732	0.9588	0.9328
Test11	0.8539	0.8774	0.9495	0.9195
Test12	0.8914	0.9284	0.9758	0.9268
Test13	0.8265	0.7770	0.9313	0.8980
Test14	0.5984	0.7893	0.8531	0.8027
Test15	0.8184	0.8244	0.9481	0.9048
Test16	0.8576	0.8798	0.9477	0.9186
Test17	0.8831	0.9159	0.9506	0.9355
Test18	0.7550	0.8498	0.8958	0.8705
Test19	0.5169	0.5956	0.8111	0.7437
Test20	0.7437	0.8258	0.9099	0.8739
Test21	0.6628	0.7646	0.8947	0.8273
Test22	0.9236	0.9391	0.9811	0.9459
Test23	0.8797	0.8793	0.9475	0.9263
Test24	0.8353	0.8953	0.9598	0.9069
Test25	0.8405	0.8475	0.9743	0.8899
Average of 25 videos	0.7604	0.8190	0.9112	0.8744
Average of 20 videos	0.8111	-	-	-

A.6 ResNeSt50 + LSTM + Non-local operator

Table A.6: Test results of ResNeST50 + LSTM + Non-local operator

Videos	F1-score	Accuracy	Precision	Recall
Test01	0.2771	0.4019	0.7270	0.6294
Test02	0.2027	0.4136	0.6868	0.5851
Test03	0.8673	0.9046	0.9853	0.8998
Test04	0.8892	0.8716	0.9705	0.9237
Test05	0.9277	0.9126	0.9769	0.9506
Test06	0.7895	0.8372	0.9269	0.8644
Test07	0.4960	0.5962	0.7974	0.7344
Test08	0.8217	0.9029	0.9293	0.9029
Test09	0.8203	0.8326	0.9391	0.9074
Test10	0.8530	0.8239	0.9878	0.8935
Test11	0.8065	0.8268	0.9786	0.8591
Test12	0.8920	0.9258	0.9740	0.9291
Test13	0.7737	0.8579	0.9283	0.8637
Test14	0.6167	0.8046	0.8515	0.7891
Test15	0.8102	0.7975	0.9578	0.8704
Test16	0.8907	0.8724	0.9766	0.9223
Test17	0.7725	0.7984	0.9558	0.8516
Test18	0.8336	0.8641	0.9652	0.8941
Test19	0.4790	0.4973	0.7813	0.7308
Test20	0.7994	0.8521	0.9325	0.8911
Test21	0.6862	0.7000	0.8971	0.8129
Test22	0.7008	0.7769	0.9380	0.8102
Test23	0.8566	0.8582	0.9735	0.8984
Test24	0.7932	0.8314	0.9698	0.8645
Test25	0.8418	0.7847	0.9783	0.8865
Average of 25 videos	0.7399	0.7818	0.9194	0.8466
Average of 20 vidoes	0.7850	-	-	-

A.7 ResNet50+ LSTM + Temporal variation layer + Non-local operator

Table A.7: Test results of ResNet50 + LSTM + TVL + Non-local operator

Videos	F1-score	Accuracy	Precision	Recall
Test01	0.3227	0.4682	0.7572	0.6778
Test02	0.2561	0.4680	0.6830	0.6099
Test03	0.9109	0.9066	0.9735	0.9362
Test04	0.9079	0.8737	0.9744	0.9341
Test05	0.9378	0.9339	0.9808	0.9574
Test06	0.8025	0.8991	0.9282	0.8836
Test07	0.4881	0.5731	0.8274	0.7350
Test08	0.8538	0.9345	0.9440	0.9120
Test09	0.7520	0.7783	0.8971	0.8894
Test10	0.8016	0.7799	0.9392	0.8898
Test11	0.7312	0.8113	0.9545	0.8317
Test12	0.9224	0.9386	0.9912	0.9363
Test13	0.7634	0.8058	0.8895	0.8876
Test14	0.7061	0.8198	0.9132	0.8184
Test15	0.8361	0.8264	0.9557	0.9084
Test16	0.7500	0.7654	0.9059	0.8757
Test17	0.8093	0.8395	0.9591	0.8725
Test18	0.7857	0.7711	0.9553	0.8613
Test19	0.4229	0.5656	0.7543	0.7112
Test20	0.7874	0.8315	0.9211	0.8942
Test21	0.6180	0.6958	0.8733	0.7956
Test22	0.7803	0.7951	0.9487	0.8681
Test23	0.8414	0.8736	0.9465	0.9045
Test24	0.8791	0.8934	0.9656	0.9234
Test25	0.7936	0.8186	0.9307	0.8728
Average of 25 videos	0.7384	0.7867	0.9108	0.8555
Average of 20 videos	0.7885	-	-	-

A.8 DenseNet169 + LSTM + Temporal variation layer + Non-local operator

Table A.8: Test results of DenseNet169 + LSTM + TVL + Non-local operator

Videos	F1-score	Accuracy	Precision	Recall
Test01	0.4302	0.6021	0.7646	0.7239
Test02	0.2613	0.4626	0.6857	0.6215
Test03	0.9166	0.9304	0.9770	0.9423
Test04	0.8924	0.9089	0.9796	0.9244
Test05	0.9468	0.9574	0.9796	0.9684
Test06	0.7777	0.8938	0.9159	0.8927
Test07	0.4197	0.5875	0.7887	0.7228
Test08	0.8488	0.9345	0.9385	0.9133
Test09	0.6968	0.7767	0.8694	0.8456
Test10	0.8792	0.8820	0.9627	0.9426
Test11	0.8115	0.8833	0.9557	0.8824
Test12	0.9040	0.9412	0.9859	0.9338
Test13	0.8368	0.7932	0.9378	0.9039
Test14	0.6500	0.8020	0.8960	0.8203
Test15	0.8323	0.8285	0.9660	0.9018
Test16	0.8673	0.8900	0.9522	0.9263
Test17	0.9252	0.9354	0.9760	0.9499
Test18	0.8110	0.8698	0.9510	0.8921
Test19	0.5284	0.6694	0.7945	0.7590
Test20	0.6762	0.8090	0.8693	0.8548
Test21	0.6768	0.7479	0.8740	0.8393
Test22	0.8959	0.9148	0.9856	0.9218
Test23	0.8659	0.8602	0.9431	0.9179
Test24	0.8426	0.9089	0.9693	0.9115
Test25	0.7821	0.8441	0.9466	0.8729
Average of 25 videos	0.7590	0.8253	0.9146	0.8714
Average of 20 videos	0.80397	-	-	-

A.9 ResNeSt50 + LSTM + Temporal variation layer + Non-local operator

Table A.9: Test results of ResNeSt50 + LSTM + TVL + Non-local operator

Videos	F1-score	Accuracy	Precision	Recall
Test01	0.3712	0.4960	0.7419	0.6722
Test02	0.2105	0.4432	0.6836	0.5862
Test03	0.8875	0.8728	0.9846	0.9126
Test04	0.9090	0.9048	0.9763	0.9409
Test05	0.9454	0.9339	0.9812	0.9626
Test06	0.8119	0.9027	0.9398	0.8792
Test07	0.4971	0.6029	0.7784	0.7468
Test08	0.8400	0.9345	0.9297	0.9204
Test09	0.7577	0.7907	0.9303	0.8760
Test10	0.8427	0.8081	0.9778	0.8954
Test11	0.8828	0.9066	0.9810	0.9133
Test12	0.8900	0.9361	0.9853	0.9232
Test13	0.8176	0.8651	0.9373	0.8930
Test14	0.7021	0.8223	0.9075	0.8228
Test15	0.8403	0.8492	0.9596	0.8961
Test16	0.8700	0.8724	0.9642	0.9180
Test17	0.8575	0.8728	0.9684	0.9031
Test18	0.7573	0.8426	0.9096	0.8701
Test19	0.5723	0.5478	0.8445	0.7741
Test20	0.7781	0.8315	0.9339	0.8804
Test21	0.7461	0.7917	0.9016	0.8539
Test22	0.7425	0.7972	0.9357	0.8400
Test23	0.8842	0.8793	0.9772	0.9113
Test24	0.8209	0.8702	0.9704	0.8832
Test25	0.7972	0.8119	0.9645	0.8840
Average of 25 videos	0.7613	0.8074	0.9226	0.8624
Average of 20 videos	0.8014	-	-	-