UNIVERSITY OF SCIENCE
FACULTY OF INFORMATION TECHNOLOGY
HONORS PROGRAM

TRẦN QUỐC CƯỜNG

VIDEO OBJECT ANNOTATION WITH INTERACTIVE SEGMENTATION

BACHELOR OF SCIENCE IN COMPUTER SCIENCE

HO CHI MINH CITY, 2020

UNIVERSITY OF SCIENCE
FACULTY OF INFORMATION TECHNOLOGY
HONORS PROGRAM

TRẦN QUỐC CƯỜNG - 1612843

VIDEO OBJECT ANNOTATION WITH INTERACTIVE SEGMENTATION

BACHELOR OF SCIENCE IN COMPUTER SCIENCE

THESIS ADVISORS:
ASSOC. PROF. TRẦN MINH TRIẾT

ACADEMIC YEAR 2016-2020

COMMENT OF THESIS'S ADVISOR

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Tp HCM, ngày .... tháng .... năm 2020

Giáo viên hướng dẫn

COMMENTS OF THESIS'S REVIEWER

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Khóa luận đáp ứng yêu cầu của Khóa luận cử nhân CNTT.

Tp HCM, ngày .... tháng .... năm 2020

Giáo viên phản biện

ACKNOWLEDGEMENT

First of all, I would like to express my sincere appreciation to my thesis advisor, Assoc. Prof. Trần Minh Triết, for his research experience, thoughtful guidance, and constructive comments. Thanks to his guidance, I have gained a lot of knowledge in the field of Deep Learning and Computer Vision. He is my role model for the depth of knowledge, scientific research passion, and especially his personality. With his kindly instruction, I overcame many challenges in the process of the thesis's research.

I would also like to extend my gratitude to lecturers in the Faculty of Information Technology, University of Science, VNU-HCM. Their enthusiastic lessons help to build a solid background in Computer Science to complete this thesis.

Besides, I am thankful to my classmates and my colleagues at the SELAB, University of Science: Vũ Lê Thế Anh, Nguyễn Hải Đăng, Trần Mai Khiêm, and especially Nguyễn Hy Hoài Lâm who always give their best in helping us and providing valuable comments.

I also thank AIOZ Pte Ltd and the research group of Prof. Minh Do at the University of Illinois at Urbana-Champaign for supporting the thesis's research with the computing infrastructure.

Last but not least, my deepest gratitude goes to my parents. Their immeasurable, unconditional love has always been the source of my energy. Words cannot fully describe my gratitude for the support that they have given me.

THESIS PROPOSAL

Thesis title: Video Object Annotation with Interactive Segmentation
Advisor: Assoc. Prof. Trần Minh Triết
Duration: January 2nd, 2020 to August 30th, 2020
Student: Trần Quốc Cường (1612843)
Theme of Thesis: applied research, proposed novel method.
Content: The objective of this thesis is to propose a novel method for interactive video object segmentation that satisfies all designed goals, including being fast, generating a good initial result, and improving accuracy after interactions. Besides, we build a video object annotation tool with the proposed method is the core algorithm. The details include: • Research on the video object segmentation task and its interactive scenario, including problem statements, challenges, datasets, evaluation metrics, and applications. • Survey on different approaches for interactive video object segmentation. • Propose a novel method for interactive video object segmentation. • Experiment on different configurations and evaluate the result based on two metrics: AUC and $\mathcal{J}\&\mathcal{F}$60s. • Develop the video object annotation tool.

- Develop the web-based platform that shows the result of the annotation tool.

Implementation plan:

- January 2nd 2020 - February 15th, 2020: Learn fundamental knowledge about Deep Learning and Computer Vision: the architecture of Neural Network and Convolutional Neural Networks.
- February 16th, 2020 - February 29th, 2020: Learn about the video object segmentation task and its interactive scenario, including problem statements, challenges, datasets, evaluation metrics, and applications.
- March 1st, 2020 - March 31st, 2020: Survey on different approaches for solving Computer Vision tasks such as image segmentation (U-Net, DeepLab, and Mask R-CNN), video object segmentation (semi-supervised approaches and unsupervised approaches), and its interactive scenario (SDI-Net, SDP-Net, and MA-Net).
- April 1st, 2020 - May 31st, 2020: Propose a novel method for interactive video object segmentation and experiment on the DAVIS 2017 dataset with two metrics: AUC and $\mathcal{J}\&\mathcal{F}$60s
- June 1st, 2020 - July 15th, 2020: Develop the annotation tool in the Qt platform and the web-based semantic video platform.
- July 16th, 2020 – August 30th, 2020: Write and finalize the thesis.

Advisor  Assoc. Prof. Trần Minh Triết	**December 15th, 2019** **Author**  Trần Quốc Cường

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Video object segmentation aims at the pixel-level separation of the objects of interest for all frames of a video. In its interactive scenario, the annotator gives iterative refinement inputs in a user-friendly annotation form, e.g. scribbles, to the algorithm to guide the segmentation process. Specifically, the annotator can gradually refine the outputs by drawing scribbles on the falsely predicted regions.

Methods for tackling the interactive video object segmentation task have to satisfy multiple goals, including being fast, generating a good initial result, and improving accuracy after interactions. In this thesis, we propose a novel method that comprises three modules: Control-point-based Scribbles-to-Mask, Self-Reference Refinement, and Guided Mask Propagation. It also maintains for each video a memory pool containing reliable frame-mask pairs, which is global across interactions, called Global Memory Pool. We conduct the experiments on the DAVIS 2017 dataset. Our proposed method achieved 0.767 and 0.759 in terms of Area Under the Curve (AUC) and $\mathcal{J}\&\mathcal{F}$ at 60 seconds ($\mathcal{J}\&\mathcal{F}$@60s), respectively, ranking 2^{nd} in the Interactive Scenario of The DAVIS Challenge 2020.

There is a lack of annotation tools for video object segmentation. In most cases, annotators have to annotate all frames in the video. In this thesis, we propose a video object annotation tool by applying interactive video object segmentation. By using this tool, annotators do not have to annotate all frames but only high-information pivot frames. Besides, pixel-level annotation is not required. This technique extends the ability to build video object segmentation datasets that contain videos of arbitrary length. The pixel-level segmentation annotation increases the interoperability of video. By embedding semantic information, we build a web-based platform that shows the object's information when interacting with their segments in the video.

CHAPTER 1
INTRODUCTION

In this chapter, we present an overview of video object segmentation and its interactive scenario, including problem statements, challenges, and applications. The motivation and objectives of this thesis are also discussed. Besides, in this chapter, we also show an overview of the thesis's structure.

1.1 Overview

1.1.1 Video Object Segmentation

The goal of computer vision is to simulate the human visual system to gain a high-level understanding of videos and images. The rapid development of smartphones, camera devices, and the social network has led to an exponential increase in media data. There is an urge to extract and analyze information from these data. Video object segmentation is the task that generates pixel-level segment regions of objects and is used for gaining fine-grained information from videos. Video object segmentation is a fundamental task of computer vision and is the core algorithm for behavior recognition and video retrieval.

Video object segmentation aims at separating foregrounds of the objects of interest for all frames in a video sequence. The goal is pixel-level tracking all objects in a video. Video object segmentation is a fine-grained task that requires methods that have to handle interacting objects, occlusion, deformation, motion blur, scale variation. Tracking objects through frames also have to handle challenges such as fast motion, out-of-view.

Depend on the input that video object segmentation is separated into two categorized: semi-supervised (human-guided) and unsupervised (no manual anno-

tation). In the former scenario, as can be seen from Figure 1.1, the mask of all objects in the first frame is provided. In the latter scenario, as can be seen from Figure 1.2, no annotation is provided.

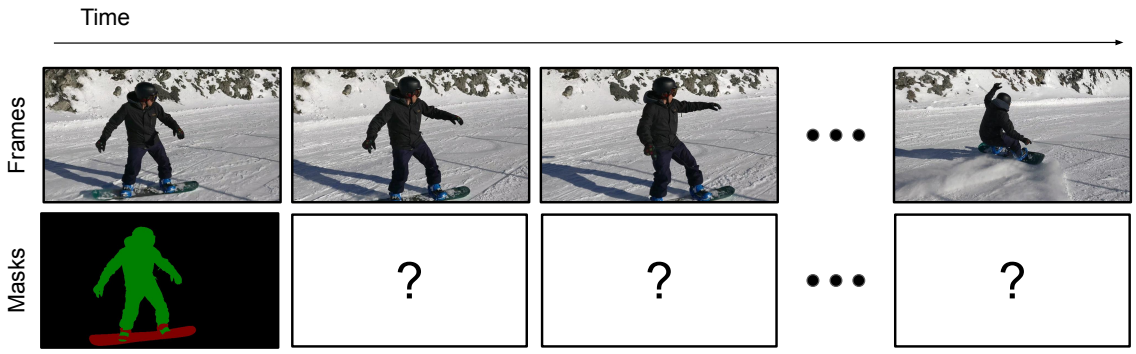

Figure 1.1: Setting of semi-supervised video object segmentation.

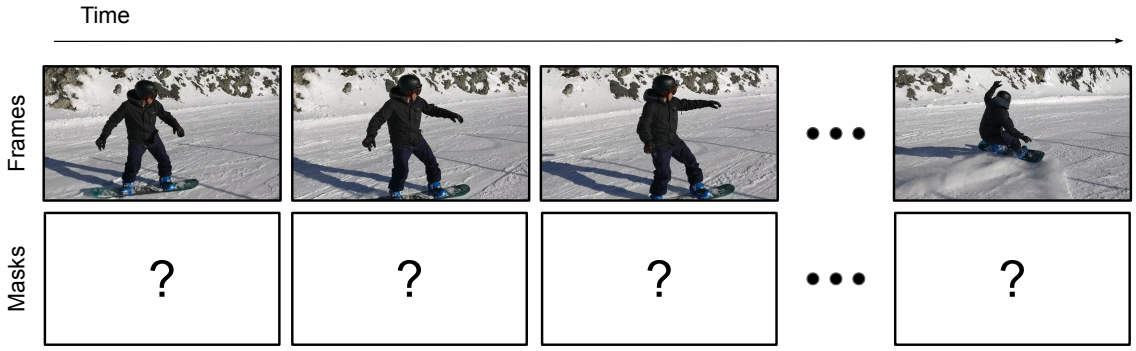

Figure 1.2: Setting of unsupervised video object segmentation.

1.1.2 Interactive Video Object Segmentation

In the interactive scenario, the annotator gives iterative refinement inputs to the algorithm to segment the objects of interest. During the first interaction, the annotator chooses a arbitrary frame and providing a user-friendly annotation form, e.g., scribbles for each object in this frame. Based on these scribbles, the model

has to predict segmentation masks of those objects for the annotated frame and temporal propagation to all frames in the video. In the next interaction, the annotator chooses a frame with the worst accuracy and provides a new set of scribbles in this frame. These scribbles point out false positive and false negative regions. The model uses these scribbles to refine its previous prediction masks. This procedure is repeated until the annotator is satisfied with the segmentation mask of all the frames.

Figure 1.3 illustrates the mechanism for the interactive scenario of video object segmentation.

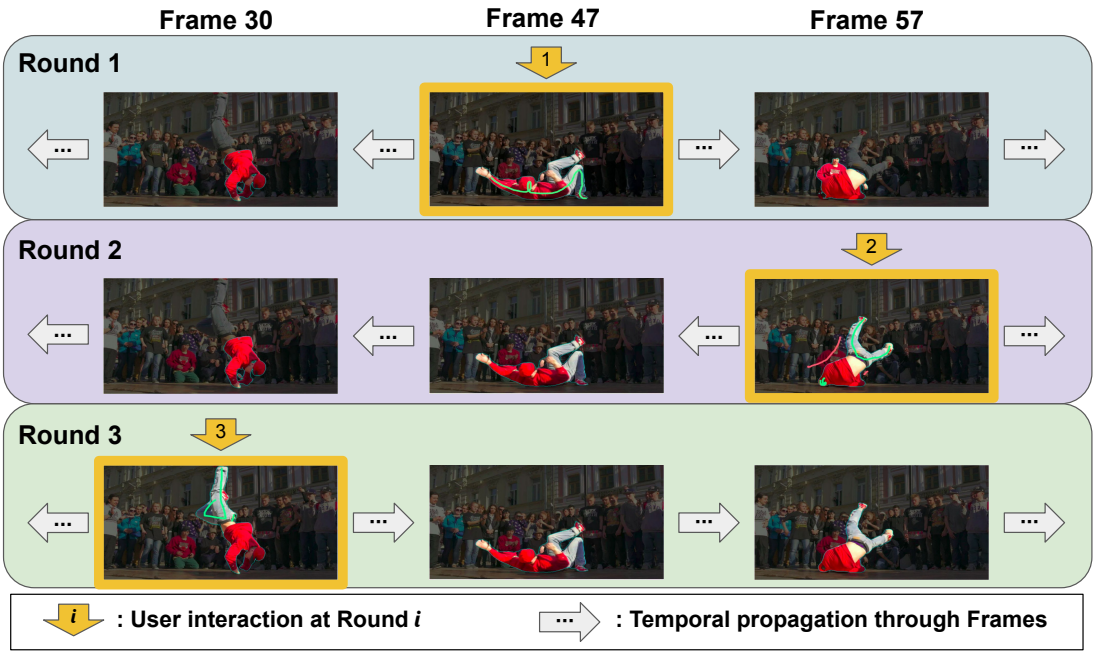

Figure 1.3: Round-based interactive video object segmentation.

Methods for tackling interactive video object segmentation task have to satisfy multiple goals, including being fast, generating a good initial result, and improving accuracy after interactions.

1.2 Motivation

Video object segmentation benefits for many applications such as video retrieval, behavior recognition, video summarization, high definition video compression, human-computer interaction, and autonomous vehicles.

But compare with other fundamental tasks of computer vision, there is a lack of annotated datasets for video object segmentation task, mostly because the cost of the annotation is too much. In the supervised scenario, the annotator has to time-consuming pixel-level annotate all objects of interest in the first frame. It takes about 79 seconds per instance for pixel-level annotation in the COCO dataset [6]. In the unsupervised scenario, no user annotation is provided. Therefore, models cannot easily point out the objects of interest or separate overlap objects which are very important in multi-object cases. Especially, two scenarios lack the low-quality segmentation mask correcting mechanism.

To overcome this challenge, interactive video object segmentation is proposed as a middle-ground solution. Instead of pixel-level annotation, the annotator points out the objects of interest by a friendly annotation form e.g. scribbles. Besides, there are multi-round video interactions between annotators with the system to fix worst predicted segmentation masks and improve overall accuracy.

There is a lack of annotation tools for video object segmentation. In most cases, annotators have to annotate all frames in the video. Therefore, the datasets evaluated for methods tackling video object segmentation are dominated by short videos. By applying interactive video object segmentation to build an annotation tool for video object segmentation, annotators do not have to annotate all frames but only high-information pivot frames. Besides, pixel-level annotation is not required. This technique extends the ability to build video object segmentation datasets that contain videos of arbitrary length.

Additionally, the pixel-level segmentation annotation increases the interoperability of video. By embedding semantic information, we can build a platform for educations that kids can learn about object's information when interacting with their segment regions in the video.

Deep learning projects usually rely on multiple libraries. Running several deep learning libraries and multiple versions for a single machine without any dependency conflict is very challenging. Container technology e.g. Docker provides a solution by packing all sources code and dependencies into a single container that is portable and easy for deployment. It also increases scalability and reproducibility for deep learning projects.

1.3 Objectives

The objective of this thesis is to propose a novel method for interactive video object segmentation that satisfies all designed goals, including being fast, generating a good initial result, and improving accuracy after interactions. Besides, we build a video object annotation tool with the proposed method is the core algorithm. The main contribution of this thesis includes:

- Propose a Control-point-based Scribbles-to-Mask module that generates image segmentation mask from scribbles.
- Propose and experiment strategies of using frames from the previous interactions to generate a Global Memory Pool for yielding the mask of a query frame.
- Refine a segmentation mask by using itself as reference combine with the aforementioned Global Memory Pool.
- Propose a propagation strategy for interactive video object segmentation that frames rely only on its nearest annotated frame.

- Implement a guidance mechanism in the propagation phase to focus on the potential regions of objects.
- Develop a video object segmentation annotation tool that reduces the human effort and supports the ability to generate annotations for videos of arbitrary length.
- Develop a semantic video website that shows the result of the annotation tool. On this website, pixel-level segmentation masks of objects are embedded with metadata.
- Dockerize all applications into a Docker image that easy for reproduction and scaling.

To fulfill our objective, the detailed work that we have done in this thesis includes:

- Study fundamental knowledge about Deep Learning and Machine Learning. We also understand the architecture of neural networks, including feedforward and backpropagation phase, activation functions. Convolutional Neural Networks are variants of neural networks that widely used for solving Computer Vision problems also are reported.
- Survey different methods for solving computer vision tasks such as image segmentation, video object segmentation, and the interactive scenario of video object segmentation.
- Study the dataset, interactive framework, and evaluation metrics for interactive video object segmentation.
- Propose a method that satisfies goals, including being fast, generating a good initial result, and improving accuracy after interactions

- Conduct experiments with the proposed method.
- Study about frameworks that need to create the annotation tool and the semantic website, including Flask, Gunicorn, Javascript, Docker.
- Propose and realize prototype for the applications, including the video object segmentation annotation tool and the semantic website.

1.4 Thesis Content

This thesis is structured into 7 chapters:

Chapter 1

In chapter 1, we present an overview of video object segmentation and its interactive scenario, including problem statements, challenges, and applications. The motivation and objectives of this thesis are also discussed. Besides, in this chapter, we also show an overview of the thesis's structure.

Chapter 2

In chapter 2, we describe relevant background knowledge about Machine Learning, Neural Network. The architecture, advantage and disadvantage of fundamental networks are shown in this chapter. We also introduce Convolutional Neural Networks and their architectures. They are the backbone of our model's modules for extracting features from the input image.

Chapter 3

In chapter 3, recent state-of-the-art approaches that solve computer vision tasks are introduced, including image segmentation, video segmentation in semi-supervised and unsupervised scenarios. We also deepen in approaches for interactive video object segmentation. At last, we show models that our method's modules rely on.

Chapter 4

In chapter 4, we present our proposed method for interactive video object segmentation. Our proposed method consists of three modules: Control-point-based Scribbles-to-Mask, Self-Reference Refinement, and Guided Mask Propagation. At first, the pipeline of the method is shown. Then, each module is introduced and the detailed implementation is presented.

Chapter 5

In chapter 5, we describe the dataset, evaluation platform, and evaluation metrics. Besides, we experiment with different configurations of our proposed method. With that, we also analyze the result and report observations. At last, the result of our proposed method in the Interactive Scenario of The DAVIS Challenge 2020 is shown.

Chapter 6

In chapter 6, we present the applications of our proposed method for interactive video object segmentation, including a video object segmentation annotation tool and a semantic video website. The layout and features of each application are shown in this chapter.

Chapter 7

In chapter 7, we report the results in the process of the thesis's research. We also draw the future works for improving our proposed method and its applications.

CHAPTER 2
BACKGROUND

In this chapter, we describe relevant background knowledge about Machine Learning, Neural Network. he architecture, advantage and disadvantage of fundamental networks are shown in this chapter. We also introduce Convolutional Neural Networks and their architectures. They are the backbone of our model's modules for extracting features from the input image.

2.1 Machine Learning and Deep Learning

From the time when programmable computers were invented, people always wonder if it can become intelligent. The primary goal of Artificial Intelligent (AI) was to create computational models that have the ability to perform intellectual tasks such as decision making, problem-solving, and understanding human communications. Today, AI is an active research topic, produce a massive amount of practical applications, and becomes the backbone of the Fourth Industrial Revolution.

The term AI was founded in the 1950s. In the early years, AI models are just straightforward commands. AI was used to solve these problems by solutions that can be defined by formal, mathematical rules. There was a period that many research in AI was reduced in funding and interest, called the AI Winter. Research results did not create sufficient breakthroughs and they were followed by disappointment and criticism. Since the 2010s, with many advancements in computer hardware, AI (especially Machine Learning) has achieved various leaps and once again proved to be noteworthy for a dramatic increase in funding and investment.

Computer Vision is a large field where most of the achievements in Machine

Learning take place. Computer Vision and Machine Learning have become very analogous nowadays. Along with Machine Learning, computational models can understand more abstract features in images from that complement the performance of Computer Vision tasks. Examples of these are K-Mean Clustering, which is a widely known algorithm for image clustering; SIFT [7] and SURF [8] are used in many applications that require a feature extractor.

Deep Learning is a subset field of Machine Learning that focuses on further improving the accuracy of computational models by manipulating the models, such as increasing its complexity hence requires more computational cost. With the development of AI-accelerated hardware (specifically GPU and TPU) and ever-increasing amount of data, Deep Learning models can understand even more abstract and complex feature patterns of the dataset. Deep Learning models outperformed human-level accuracy in many fields such as object detection, voice generation, and prediction.

2.2 Neural Network

2.2.1 Overview

The origin of artificial neural networks (ANN) is when Warren McCulloch, a neurophysiologist and Walter Pitts, a mathematician wrote a paper [9] how a computational model for neural networks based on algorithms called threshold logic. The goal of a neural network is to approximate a function f^* that maps from an input domain to an output domain. A neural network is a stack of layers, including an input layer, followed by hidden layers, and finally an output layer. Each layer consists of a number of nodes, called units or neurons. Between the nodes of two consecutive layers, certain links are created to transfer the value from the input layer to the output layer and vice versa. Figure 2.1 shows an example of a neural network. Neural networks are used for extracting patterns

of a collection of inputs and are the quintessential deep learning models.

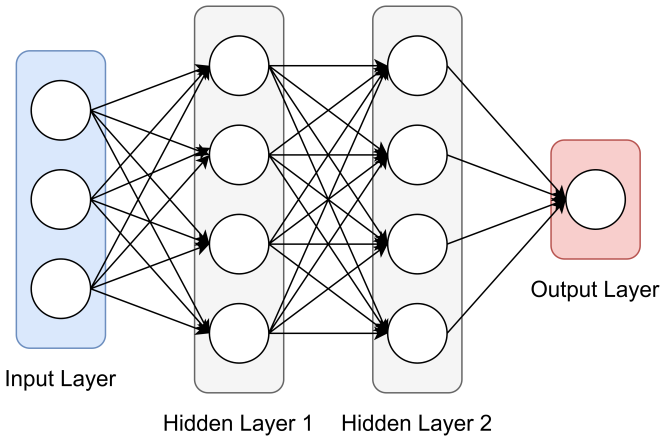

Figure 2.1: Architecture of a neural network with two hidden layers.

2.2.2 Feedforward

Essentially, the learning process in a neural network consists of two phases, namely feedforward and backpropagation.

A feedforward and a backward propagation operation took place on a dataset is called an epoch. Over the span of an epoch, the network becomes better fitted to the data that was fed into it, by adjusting its weights according to the feedback from the backpropagation operation, which itself was derived from the output of the feedforward operation. This effectively generalizes the network to the pattern of trained data over time.

Feedforward helps create a flow and passes the data from the input layer, throughout hidden layers, to the output layer.

Figure 2.2 describes the feedforward process. In the figure, the neural network has 3 layers. The circles labeled "+1" in the first two layers are called "bias unit".

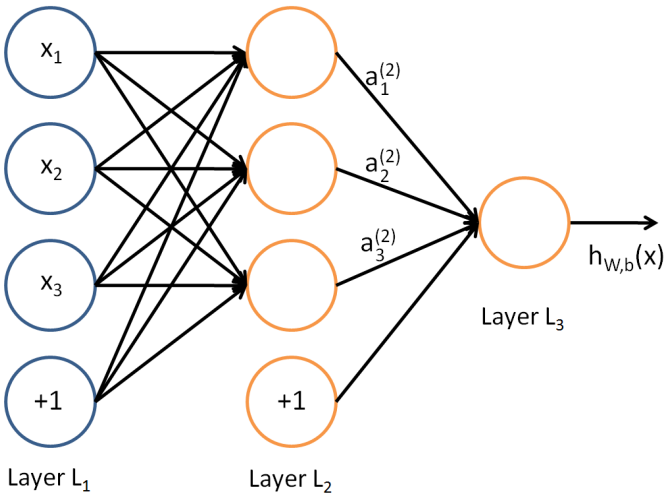

Figure 2.2: Feedforward in neural network [1].

Given $W_{ij}^{(l)}$ is the weight associated with the connection between unit j in layer l and unit i in layer $l+1$. In this example, we have $1 \leq l \leq 2$ and the number of units in layers are 3, 3, and 1 , respectively.

Given $a_i^{(l)}$ is the output value (the value after mapping from activation function) of unit i in layer l. Activation functions are non-linear, differentiable functions, denoted by $g(z)$. These will be explained in Section 2.2.4. In the input layer, $a_i^{(1)} = x_i$. The feedforward process for a hidden layer is given by:

$$
\begin{aligned}
a_1^{(2)} &= g(W_{11}^{(1)}x_1 + W_{12}^{(1)}x_2 + W_{13}^{(1)}x_3 + b_1^{(1)}) \\
a_2^{(2)} &= g(W_{21}^{(1)}x_1 + W_{22}^{(1)}x_2 + W_{23}^{(1)}x_3 + b_2^{(1)}) \\
a_3^{(2)} &= g(W_{31}^{(1)}x_1 + W_{32}^{(1)}x_2 + W_{33}^{(1)}x_3 + b_3^{(1)})
\end{aligned}
$$

Given $h_{W,b}(x)$ is denote the output value of neural network with parameters W, b and the input x, the value of $h_{W,b}(x)$ is represented by the following equation:

$$
h_{W,b}(x) = a_1^{(3)} = g(W_{11}^{(2)}a_1^{(2)} + W_{12}^{(2)}a_2^{(2)} + W_{13}^{(2)}a_3^{(2)} + b_1^{(2)})
$$

2.2.3 Backpropagation and Gradient Descent

Optimization problems are the problems of finding the best solution (global maximum or global minimum) from all available solutions. However, in most cases, the cost of finding the global maximum (minimum) is too much, or sometimes impossible. Gradient descent is an iterative optimization algorithm that finds a local maximum (minimum) by taking multiple steps from the current solution to better ones. In machine learning, based on how much data was feed in an iteration, gradient descent is divided into 3 variants: batch gradient descent, stochastic gradient descent, and mini-batch gradient descent. There is a trade-off between the accuracy of each iterative and computing time, depending on the problem at hand.

In machine learning, gradient descent tries to optimize the accuracy by finding the local minimum of the loss function J. A loss function is a function that measures the distance between prediction and the ground truth. With the loss value of current prediction, based on the gradient value of J, the parameters at the output layer will be recalculated. In the opposite direction with feedforward, the new gradient value will be propagated to the previous layers in order to update the parameters in these layers. Based on this principle, all the parameters in the network are updated and the feedforward process will continue its next iteration in order for the neural network to find better parameters that are more fit to the data and yields better accuracy results.

2.2.4 Activation Functions

A problem that comes up when designing neural networks is choosing the activation function to use in hidden layers of the model. Activation functions control a flow that presented data in the current hidden layer and produces output through some gradient processing, besides a loss function that design to estimate

the error of approximation to make neural networks learnable. For each hidden unit in hidden layers, the process of computing the value can be described as the following: The input vector x is passed to an affine function $z = W^T x + b$ then a non-linear activation function $g(z)$. Research related to the impact of activation function always draw attention from the research community. In experiments, it demands a lot of trial and error to choose the best activation function for the designed model.

2.2.4.1 Sigmoid Function

The formula of the Sigmoid function is defined as below:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

The Sigmoid function maps real-values to values in the range of 0 to 1. As can be seen from Figure 2.3, with a very high value of z, then the value of $e^{-z} \approx 0$, therefore $\sigma(z) \approx 1$. With a very low value of z, then the value of e^{-z} will be very high, therefore $\sigma(z) \approx 0$.

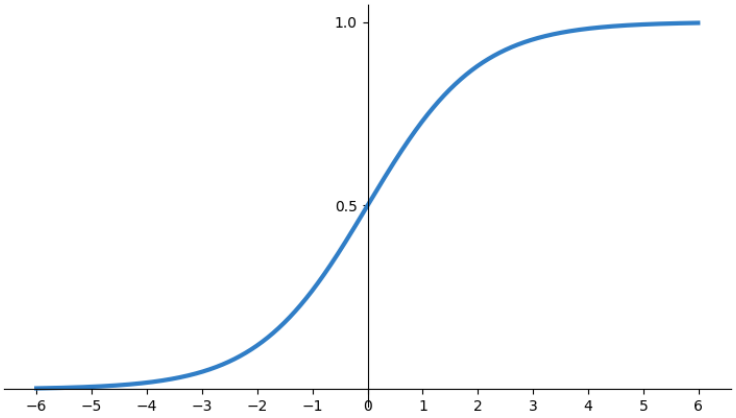

Figure 2.3: Sigmoid function.

Since the Sigmoid function is not zero-centered, in case all the input values x have the same sign (positive or negative), the gradient $\frac{dJ}{dW_i}$ always has the same sign as $\frac{dJ}{dz}$. This makes the learning process often overshoots and result in a zig-zag path to the optimal point.

2.2.4.2 TanH Function

Unlike the Sigmoid function, the TanH function normalizes real-value to the range of -1 to 1.

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

The TanH function can be transformed into the Sigmoid function as follows: $tanh(z) = 2\sigma(2z) - 1$.

Figure 2.4 shows the plot of the TanH function. TanH function is zero-centered.

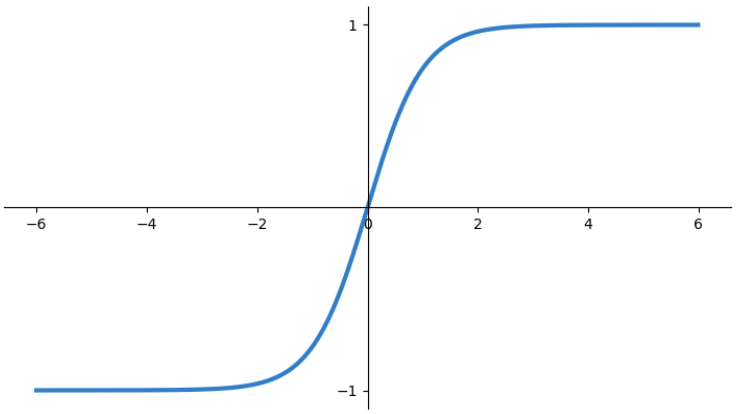

Figure 2.4: TanH function.

Both the Tanh and the Sigmoid have the same disadvantages. They are only sensitive to their input when the value z is near 0. If z is too high, or too low, there is almost no change to these activation function outcomes. This makes the

gradient-based learning process took more time to achieve the best prediction. Both of these functions are also computationally expensive compared to other activation functions.

2.2.4.3 Rectified Linear Unit (ReLU) Function

Rectified Linear Unit Fuction (ReLU) is defined as:

$$ReLU(z) = max(0, z)$$

ReLU is very simple and easy to compute compared with Sigmoid and TanH, because half of its image domain is identity, and the other is constant. The gradient value range will be large and also consistent. Figure 2.5 plots the ReLU function.

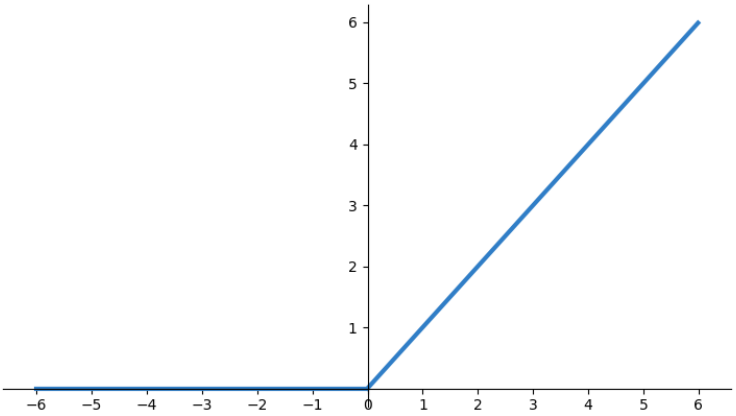

Figure 2.5: ReLU function.

Although the ReLU fucntion is not differentiable when the input value is zero, there are still sub-derivatives employed that allow for backpropagation. In the scenario where all the input values are zero or negative, the gradient of function

also becomes zero, therefore the model cannot learn. This situation is called "Dying ReLU". An extra data preprocessing step is needed to overcome this problem.

ReLU has proven its effectiveness in experiments. Many neural networks use ReLU as the default choice for activation functions of hidden units.

2.2.4.4 Leaky ReLU (LReLU) Function

Leaky ReLU is a variant of ReLU that prevents the "Dying ReLU" situation when input data are negative.

$$LReLU(z) = max(\alpha z, z)$$

Usually, α is a small positive value (about 0.01 to 0.1). The plot of Leaky ReLU is shown in Figure 2.6.

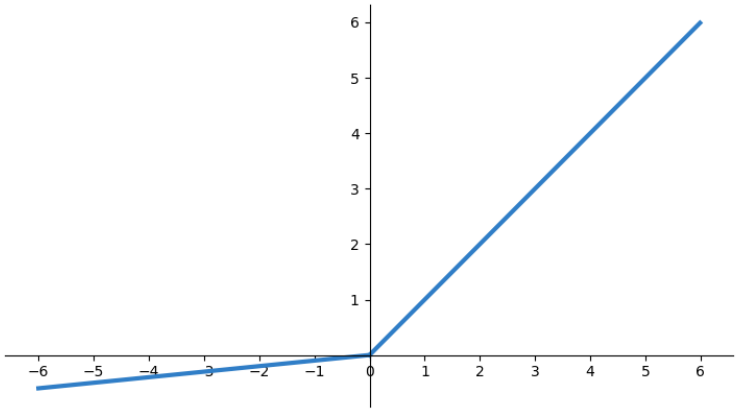

Figure 2.6: Leaky ReLU function ($\alpha = 0.1$).

2.2.4.5 Maxout Unit Function

Maxout unit is a variant of ReLU function that generalizes ReLU even further than Leaky ReLU. Instead of depending on the value of an element-wise function $g(z)$, the maxout is designed to return the maximum value of several learned parameters, defined as:

$$g(z)_i = \max_{j \in G^{(i)}} z_j$$

Where $G^{(i)}$ are the indices of the group i's inputs. This function presents a method than learning a piece-wise linear function that can describe to multiple directions of inputs.

Most of ReLU generalizations perform the same as ReLU in most cases, and occasionally outperform the latter.

2.3 Convolutional Neural Network

Computers see an image as an array of pixels. Given H, W is the image width and height, respectively. The dimension of the array will be (H x W x 3) for an RGB image, while (H x W x 1) is the dimension of a gray image.

Convolutional Neural Network (ConvNet, or CNN) is very similar to ordinary neural networks with a stack of layers and learnable parameters. It is a neural network that is designed to process grid-like data. Examples include time-series data (1D grid at time intervals), image data (2D grid of pixels). Since its invention, ConvNet has become a foundation of modern computer vision. The name Convolutional Network comes from its linear operation *convolution*. With the assumption the inputs are grid-like data, ConvNet has a grid-like topology architecture that helps reduce the number of parameters in the network.

ConvNet has created a massive impact in the field of Computer Vision research, since it allows invariance to be present in affine transformations of images, such

as rotation, scale, and translation. This is possible by the design of the Convolutional layer. In the learning phase of a ConvNet, the weight values in filters of each pixel are learnable. In addition to the width and height dimensions of each Convolutional layer, there is also a depth component. If each segment of the output of the Convolutional layer is partitioned, the obtained result will be 2D planes of a feature map. The weights of filter used on a single 2D plan are shared between pixel filters. This is one of the advantages of CNN that helped to reduce a vast amount of parameters and increasing computational efficiency.

2.3.1 Architecture

Ordinary neural networks receive a single input vector, transform between hidden layers, and pass to the output layer at last. Each hidden layer is a set of hidden units and fully connected with units in the previous layer. Take an example on the CIFAR-10 image set, the input image size is 32 x 32 x 3. In an ordinary neural network, the number of connections between one hidden unit of the first hidden layer to the input layer will be 32 x 32 x 3 = 3072. These numbers are relatively lightweight for a modern processor. However, this network cannot perform well when the input image size is much larger. If the input size becomes 200 x 200 x 3, the number of weights would total in 120,000 for just one unit. These neural networks are computationally expensive and overfit easily.

Unlike ordinary neural networks, ConvNet takes advantage that the input data has a relative relationship between parts in a grid. Each ConvNet layer has units arranged in 3 dimensions: height, width, and depth. With the CIFAR-10 image set, the dimension of the input layer is 32 x 32 x 3, and the dimension of the output layer is 1 x 1 x 10 (10 is the number of classes that the network needs to classify).

The architecture of a classical ConvNet - LeNet-5 is shown in Figure 2.7. A ConvNet consists of 3 main types of layers: Fully-Connected layer, Convolution layer, and Pooling layer. We go into details of each type of layer in the following sections.

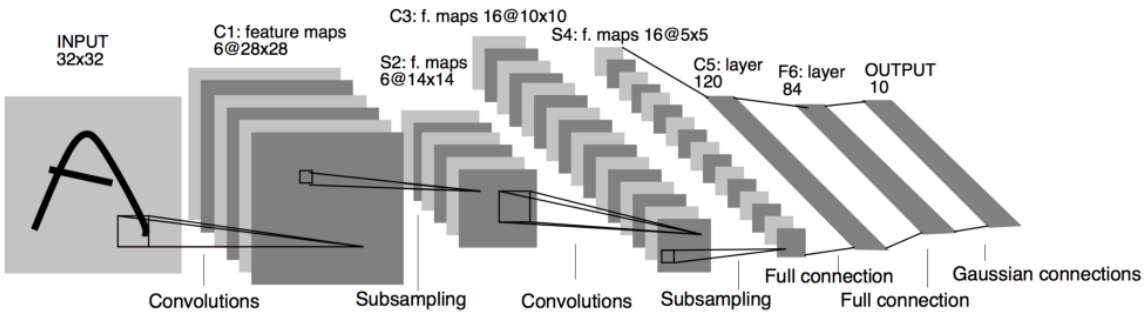

Figure 2.7: Architecture of LeNet-5 [2].

2.3.2 Fully Connected Layer

Units in the Fully-Connected layer have full connections to all activations in the previous layer. This kind of layer is very popular in ordinary neural network architectures. It is usually placed at the output stage for generalizing and summarizing features extracted from previous layers to produce the final output. Especially, in classification problems, Fully-Connected layer is used in the last layer as a probabilistic classifier where the number of units is often small and the added computational cost is negligible.

2.3.3 Convolutional Layer

The Convolutional layer is the quintessential ConvNet.

A filter is a small matrix that consists of weights. When applied to images, filters can blur, sharpen, or emboss a region of the image, depending on filter parameters. The Convolutional layer parameters are a set of learnable filters.

Filters in Convolutional layers extend with the depth of the activations of the previous layer.

During the feedforward phase, each filter slides across the width and height of the input vector and compute the dot products between the filter and the value at any cell in the input grid. The result after applying each filter is a 2-dimension feature map which each cell responses to the feature of a region in the input vector, that has the same size as the filter. It is expected that by using ConvNet, each layer can learn a type of visual features, such as edge, color, and patterns. 2-dimension feature maps from filters are stacked to create a 3-dimension vector. This vector will be passed to an activation function to create the input vector for the next layer. An example of a unit in a Convolutional layer only connects with a region in the previous layer is shown in Figure 2.8.

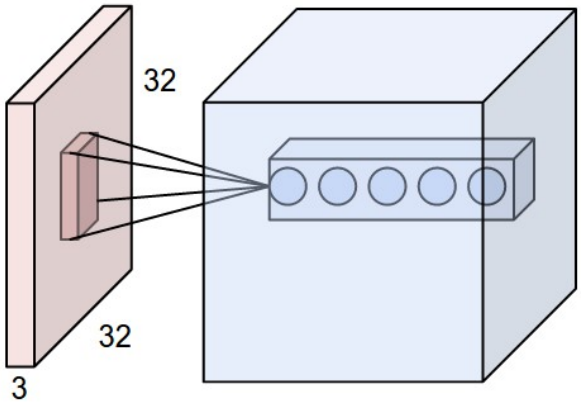

Figure 2.8: Connection between a unit in a Convolutional layer and a region in the previous layer [3].

Spatial arrangement: These number of hyperparameters that decide the size the output vector are depth, stride, and zero-padding:

- The **depth** of the output vector is a hyperparameter. Depending on the problem, choosing each depth size has its own advantage and disadvantages,

and requires trial and error in experiments.

- The **stride** decides the step when the filter slides through the input vector. The default stride is 1, which corresponds to every element of the input vector will be slided over. If the stride is equal or larger than 2, the filter will skip the proportional amount of elements.
- The **Zero-padding** is used when filters slide around the border of the input vector.

Given W, F, S, and P is the input size width, the filter size, the stride, and the zero-padding on the border, respectively. The width of the Convolutional layer is approximated by the formula:

$$\frac{W - F + 2P}{S} + 1$$

Similarly, given H is the input size height. The height of the Convolutional layer is approximated by the formula:

$$\frac{H - F + 2P}{S} + 1$$

Parameter Sharing: Compared with Convolutional layer width and height, the filter size is usually much smaller. The number of parameters is vastly reduced by sharing parameters between filters. Convolution Network takes less time for a feedforward and backpropagation iteration, therefore reduces the computational cost.

2.3.4 Pooling Layer

The Pooling layers are in charge of down-sampling the size of the input volume. Pooling is applied in each channel independently to extract features in the region

of input volume defined by the size of the window.

Pooling layers extract more generalized features and are responsible for preventing image invariance. Pooling layers also prevent overfitting by providing an abstracted form of the representation, reducing the complexity of the model. Several well-known types of pooling are Max Pooling, Average Pooling, and L2-Pooling. The difference between these types are the function that maps from the input values in the window size to the output value. Figure 2.9 shows an example of Max Pooling.

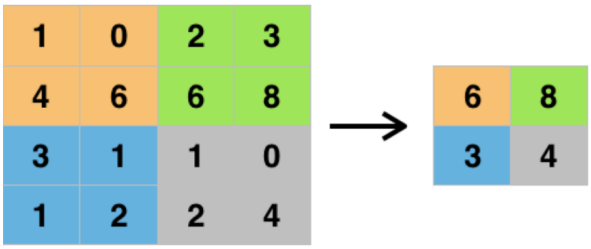

Figure 2.9: Max Pooling from a vector has size (4 x 4) to a vector has size (2 x 2) [4].

CHAPTER 3

RELATED WORKS FOR INTERACTIVE VIDEO OBJECT SEGMENTATION

In this chapter, recent state-of-the-art approaches that solve computer vision tasks are introduced, including image segmentation, video segmentation in semi-supervised and unsupervised scenarios. We also deepen in approaches for interactive video object segmentation. At last, we show models that our method's modules rely on.

3.1 Image Segmentation

Image segmentation is the task of class prediction for every pixel of an image. Depend on whether the object instances need to determined or not, image segmentation separated into two tasks: instance segmentation and semantic segmentation. Figure 3.1 shows examples of instance segmentation and semantic segmentation.

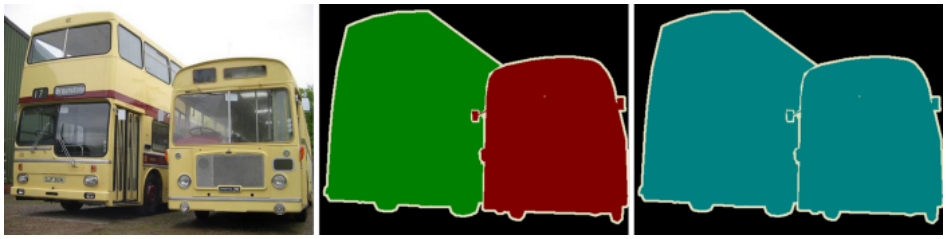

Figure 3.1: The left image: input. The middle image: the output of the instance segmentation. The right image: the output of the semantic segmentation. [5]

3.1.1 Semantic Segmentation

ConvNet alone is not suitable for this task because its layers are designed to reduce the feature dimension and producing highly decimated features. Networks that handle this task usually has two phases. The first phase is downsampling to

reduce the spatial dimensions of feature maps and learn deeper features of the input. The second phase is upsampling to recover the input features dimension size. This architecture is called Encoder-Decoder and used in a lot of famous models such as U-Net and DeepLab variants.

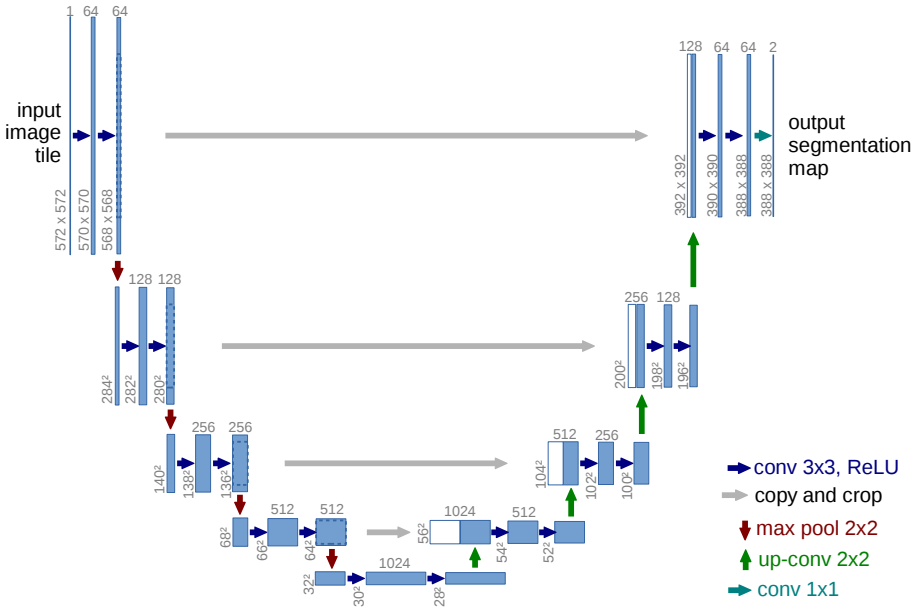

Figure 3.2: Architecture of U-Net.

U-Net [10] is designed for the biomedical image segmentation task. The architecture of U-Net is shown in Figure 3.2. In the encoder, a Convolutional layer and a Max-Pooling layer are implemented repeatedly to extract features and also reduce the dimension of feature maps. The model can learn about the "what" information in this phase. In the decoder, a Up-Convolutional layer and two Convolutional layers are implemented repeatedly to recover the dimension of feature size. Skip connections also used for concatenating the feature maps from the encoding phase with the same level, it helps to increase the localization information. The model can learn about the "where" in this phase.

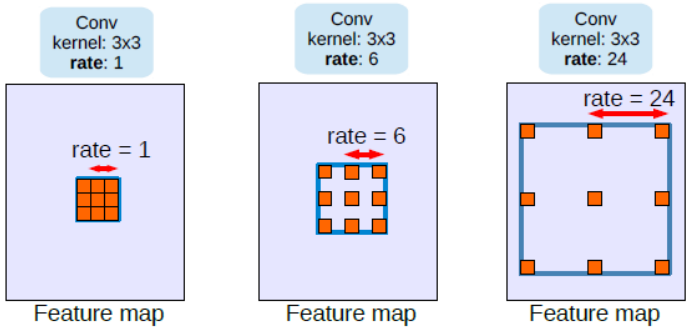

Figure 3.3: Atrous Convolution with Different Rates r.

DeepLab has a modified Encoder-Decoder architecture and used in a variety of tasks in computer vision. Atrous Convolution is first introduced in DeepLabv2 [11]. Figure 3.3 visualizes the Atrous Convolution, where rate r is the stride between input signals. With Atrous Convolution, each layer with learning features information about the larger region without increasing computational expenses. By applying this technique, we can build a "deeper" neural network. Atrous Convolutions with different rates are stacked in the encoder, called the Atrous Spatial Pyramid Pooling module. It offers the ability to learn multi-scale features in the encoding phase. In the decoder phase of DeepLabv3+ [12], encoded features upsampled by a factor of 4, after that it concatenates with the feature maps in the encoder with the same-level. Several Convolutional layers and an upsampling layer are applied at last. The architecture of DeepLabv3+ is shown in Figure 3.4.

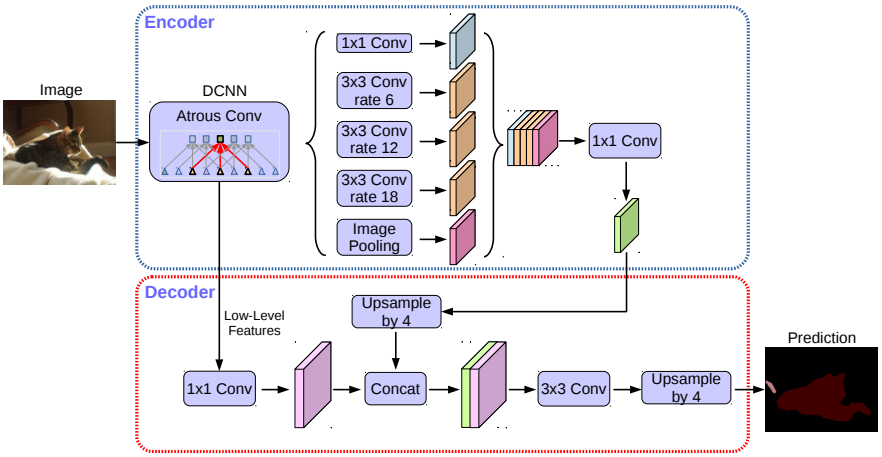

Figure 3.4: Architecture of DeepLabv3+.

3.1.2 Instance Segmentation

One of famous instance segmentation networks is Mask R-CNN [13]. It is invented by Facebook AI Research. The architecture of Mask R-CNN is shown in Figure 3.5.

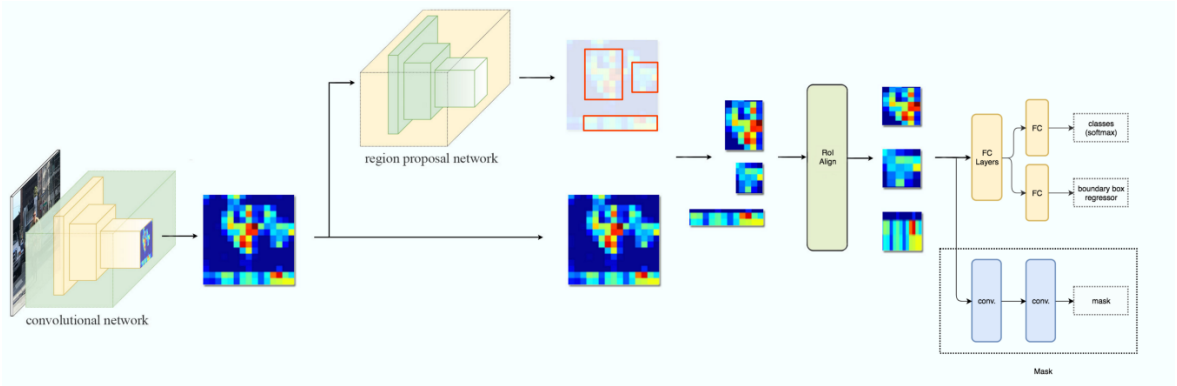

Figure 3.5: Architecture of Mask R-CNN.

Mask R-CNN consists of two stages: Region Proposal Network (RPN) and outputs generating. The input image is going through a convolutional backbone to extract features. An RPN is used for generating region proposals. The RoiPool

layer is implemented with image features wrapped by bounding box generating from the RPN module as the input. The output of the RoiPool layer is fixed-size feature maps. At last, There are two branches, a branch for instance masks generating, another branch is used for object detection.

3.2 Video Object Segmentation

Video object segmentation aims to segment objects of interest for all the frames in a video. The goal of video object segmentation is pixel-level tracking all objects in the video. Depend on whether the objects of interest are indicated or not, video object segmentation is divided into two scenarios: semi-supervised and unsupervised. In the semi-supervised scenario (human-guided), the mask of all objects of interest is provided in the first frame and requires tracking these objects in the next frames. The unsupervised scenario requires detection, segmentation, and tracking objects full automatically.

3.2.1 Semi-supervised

There are several methods based on tracking objects. An image object mask from the current frame is predicted from the RGB frame image and the masks from previous frames. FEELVOS [14] uses a pixel-wise embedding together with a global (the first frame) and a local (the previous frame) matching mechanism to transfer semantic information to the current frame. BOLTVOS [15] splits the task into two sub-tasks: bounding box level tracking, followed by bounding box segmentation for each object. PReMVOS [16] authors proposed a two-stage approach: propose a set of object masks for each frame and then select and merge these proposals into the final masks for all the frames based on tracking objects.

There is another approach based on memory networks. OSVOS [17] has a Con-

vNet learns from ImageNet [18] that in charge of foreground-background separation, and learn the appearance of each object from the first frame (online learning) and segments all frames independently. STM [19] uses previous frames and their object masks to form a memory pool for reference and uses the current frame as the query by attention mechanism.

In DAVIS Challenge [20] 2020, Tran et al. present the MR-GIS [21] framework that combines two-pass segmentation with memory-based network and object tracking with mask guidance.

3.2.2 Unsupervised

UnOVOST [22] consists of 5 phases: propose object masks for all the frames, clip object masks that they do not overlap each other, generate object tracklets by choose object proposals that consistent through a range of consecutive frames, tracklets are merged into objects by re-identification, and choose objects with the best confidence score. Shubhika Garg et al. [23] propose a method that consists of 3 stages: initial masks for each frame of the video by using Mask R-CNN, use the first frame initial mask as the reference and again generate masks for all the frames by using STM, and recover lost objects at last.

3.3 Interactive Video Object Segmentation

In the interactive scenario, the user gives iterative refinement inputs to the algorithm to segment the objects of interest.

For experiments with methods that tackle this task, the DAVIS Challenge on Video Object Segmentation has hosted the interactive scenario since 2018. During the first interaction of each sequence of frames, the server chooses a particular frame and provides a human-annotated scribble for each object in this frame. Instances of scribbles are shown in Figure 3.6. Based on these scribbles, the user's

model has to predict segmentation masks of those objects for all the frames. After that, the user submits the predicted masks to the server. In each of the subsequent interactions, from a list of frames specified by the user, the server chooses the frame with the worst prediction and provides a new set of human-simulated scribbles in this frame. These scribbles point out the false positive and false negative regions. Figure 3.7 shows examples of the subsequent scribbles, the predicted mask from the previous interaction, and provided scribbles from the server. The user's model uses these human-simulated scribbles to refine its previous prediction masks. This procedure is repeated until a maximum number of interactions or a timeout is reached. The server measures the time needed to perform each interaction, these timings are combined to compute the final results.

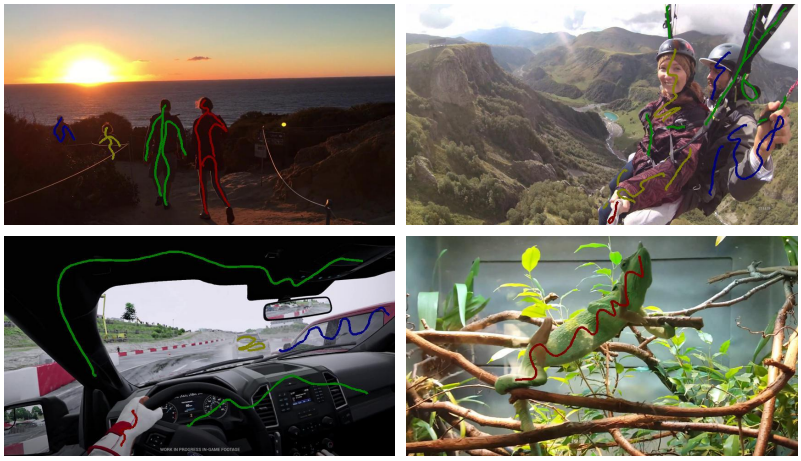

Figure 3.6: Scribbles in the first interaction.

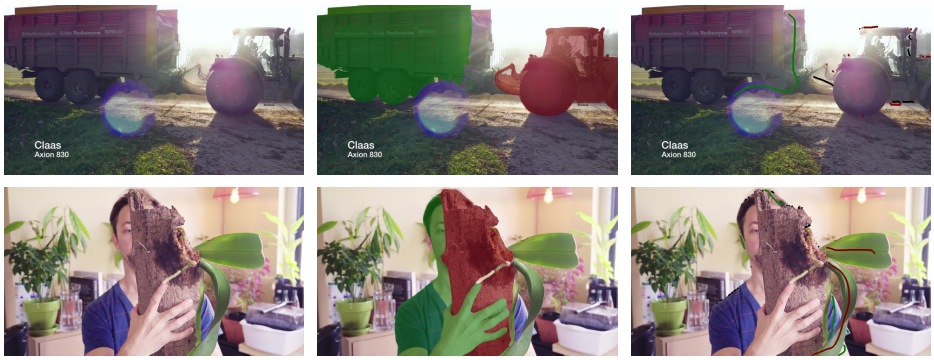

Figure 3.7: For each row, from the left most to the right most image: the frame image, the predicted mask, scribbles provided by the server.

Every year, novel methods from participants are proposed and achieved promising results. Approaches for tackling interactive video object segmentation have to satisfy some design goals, such as being fast, generating an efficiency initial video mask after the first interaction, and improving accuracy after the subsequent interactions.

A common approach for this problem consists of 2 stages: generating the image object mask from the scribbles and mask propagation. Heo et al. [24] present a model composed of 2 sparse-to-dense networks, called by SDI-Net and SDP-Net, respectively. The former network in charge of generating the segmentation mask from the scribbled frame. SDI-Net architecture is shown in Figure 3.8. SDI-Net generating the mask from the annotated frame for each object independently, these masks will be merged into the final multi-object mask for the annotated mask. There are two types of input for SDI-Net depend on whether it is the first interaction or not. SDI-Net is an encoder-decoder network. In the encoder phase, ResNet50 with skip connections is developed to extract low-level and high-level features. In the decoder phase, two parallel modules are adopted, ASPP module and bottom-up module. The probability map of the object is computed by concatenated the output of these two modules and transformed through convo-

lutional layers at last. From this mask, the method spawn points that belong to segment regions, dilate these points, and propagate dilated points to adjacent frames by using optical flow. The latter network is responsible for generating the mask for each object from these propagated dilated points. SDP-Net architecture is shown in Figure 3.9. The inputs of the model are two pairs of frame-points from the annotated frame and the current frame. At first, there are two encoders based on the Siamese structure, corresponding with two input pairs. Features extracted from two encoders are combined by convolutional layers and squeeze-and-excitation (SE) modules. ASPP module and the bottom-up module receive the combined features and the features of the second encoder, respectively. The decoder module mixer combines the output signals of two modules to yields a probability map for all objects of interest at last.

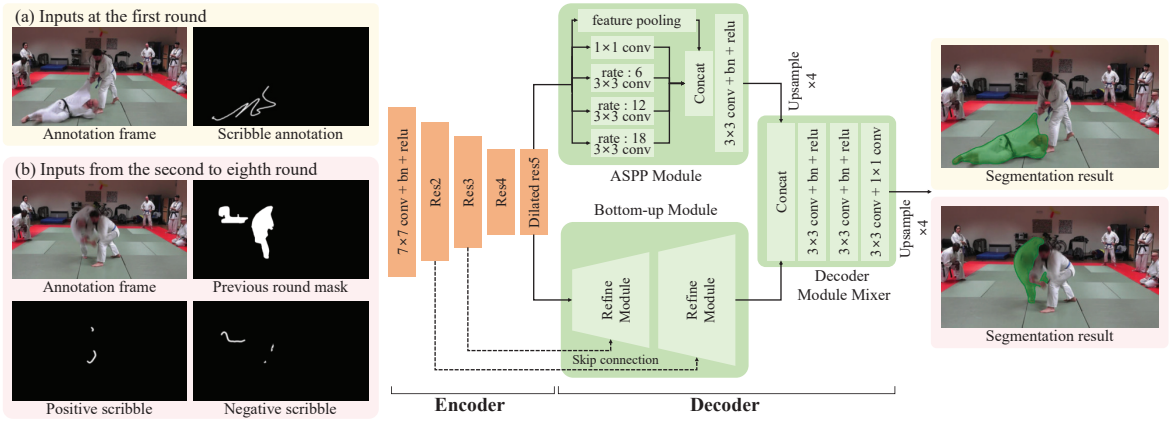

Figure 3.8: Overview of SDI-Net.

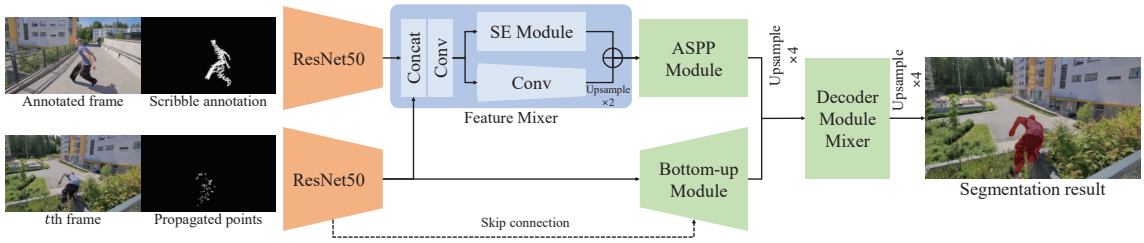

Figure 3.9: Overview of SDP-Net.

Unlike previous methods, MA-Net [25] integrates the interaction network and propagation network into a unified, pixel embedding learning framework by sharing the same backbone, which is memory efficiency. MA-Net consists of 3 modules: a pixel embedding encoder, an interaction, and a propagation branch. Figure 3.10 shows the pipeline of MA-Net. The first module - pixel embedding encoder encodes all pixels from all RGB frames of a video sequence into a pixel-wise embedding vector. The interaction branch leverages scribbles, the pixel-wise embedding vector of the annotated frame, and the segmentation mask from the previous interaction (in case it is not the first interaction) to yield the instance segmentation mask. The propagation branch propagates the informative knowledge of the user-annotated frame and the previous frame to the current frame using the pixel embedding. The pixel embedding encoders of two branches are using the same backbone and share weights with each other. As the segmentation heads, both branches have 2 "shallow" networks with several convolutional layers. The pixel embeddings of all frames are extracted exactly once time, in the first interaction round. During the subsequent interactions, only the two "shallow" segmentation heads are used leads to the network more efficient than the first interaction.

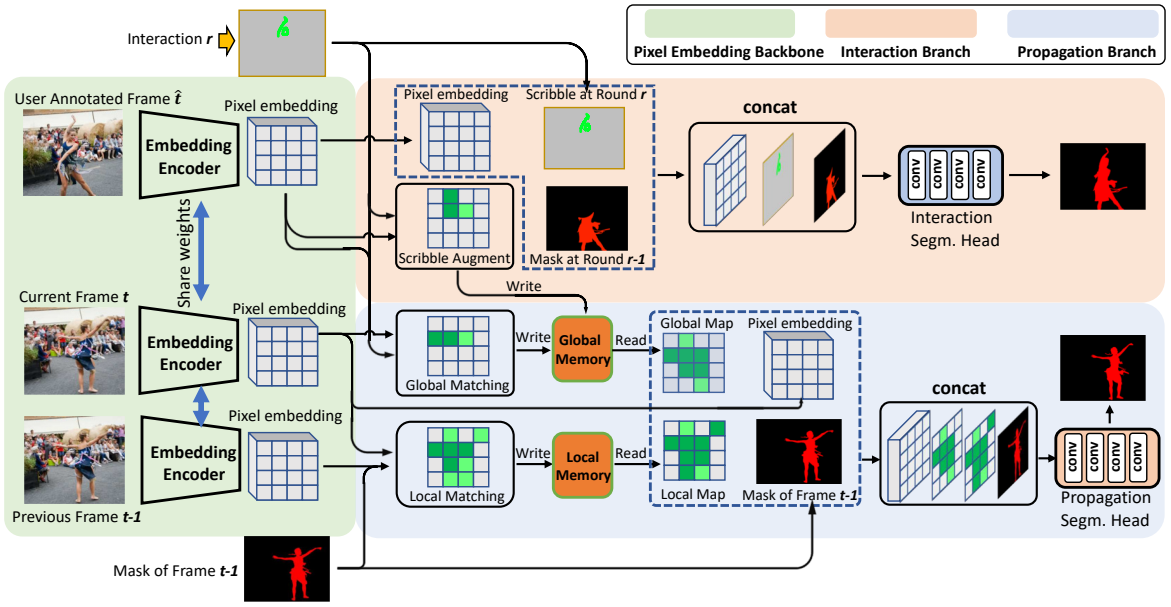

Figure 3.10: Pipeline of MA-Net.

One of the two stages of common methods in interactive video object segmentation is generating the image object mask from user-provided scribbles. By converting these scribbles into a set of control points, f-BRS [26] can be applied as an efficiency module for this stage. The other stage is mask propagation where the model has to use information from the current interaction mask in the first stage and masks from previous interactions. Memory-based models, specifically STM [19], satisfy all conditions about the accuracy and the speed for this stage.

3.3.1 f-BRS: Feature Backpropagating Refinement Scheme

The main goal of f-BRS [26] is to generate an accuracy binary image segmentation mask from a set of positive points and a set of negative points. Specifically, positive points are points in the segmentation region of the object and negative points are otherwise. By leveraging the optimization-based approach and the learning-based method, f-BRS correct disadvantages of both approaches such as lack of using semantics in the optimization-based approach case and overcon-

fidence with semantics and lack of using point coordinates information in the learning-based approach case.

Adversarial examples are instances with internal feature noise that result in an inaccurate prediction from the neural network with high confidence. Szegedy et al. [27] formalized the problem of generating adversarial examples for the classification task into an optimization problem. Given model is the function f that maps from the input to the output. $\mathcal{L}$ denotes a continuous loss function for the classification task. For a given image $x \in R^m$ and the target label l, they aim to find the nearest value of x called $x + \Delta x$ which classified as l by f. The minimization of the following energy function is formulated as:

$$\lambda||\Delta x||_2 + \mathcal{L}(f(x + \Delta x), l) \rightarrow \min_{\Delta x}$$

Where λ is a hyperparameter and serves as a trade-off between the two terms.

By formalized the interactive image segmentation task into an optimization problem, Jang et al. [28] proposed an algorithm that outperforms the conventional algorithms. The network takes the image stacked with distance maps of user-provided points. In the feedforward phase, an encoder-decoder ConvNet is proposed to output the segmentation mask of the object. But it can not guarantee that all the provided points have correct labels. Therefore, a backpropagating refinement scheme (BRS) is proposed with the goal is to find minimal edit to the distance maps that result in a consistent object mask with input points. There are two energy functions that are used: corrective energy function corresponding with the coordinates of input points and inertial energy function in charge to keep the network stable with small changes to the input. This approach requires backpropagation through the whole network. It is computationally expensive. Let $f(x)_{u,v}$ is the output of the model with the input image x and the input

point (u, v) with label l. In this task, the energy function is formulated as:

$$\lambda||\Delta x||_2 + \sum_{i=1}^{n}(f(x + \Delta x)_{u_i,v_i} - l_i)^2 \rightarrow \min_{\Delta x}$$

f-BRS reparameterizes the function f by adding the auxiliary variables for optimization. Let $\hat{f}(x, z)$ denotes the function that maps from the input x combines with the introduce variables z to the output. With auxiliary parameters fixed $z = p$ then $\hat{f}(x, p) \equiv f(x)$. The optimization problem is converted into:

$$\lambda||\Delta p||_2 + \sum_{i=1}^{n}(\hat{f}(x, p + \Delta p)_{u_i,v_i} - l_i)^2 \rightarrow \min_{\Delta p}$$

By finding the closest value of p via $p + \Delta p$, we do not have to go through the whole network for finding Δx. For efficient, f-BRS chooses the channel-wise scaling and bias for the activation at one of the last layers for reparameterization because it does not have a localized effect on the output and it just needs to backpropagate through a small part of the network for optimization. Depend on the position of the layer in which scaling and bias are applied, f-BRS can trade-off between the accuracy and the speed. Figure 3.11 shows the architecture of f-BRS variants.

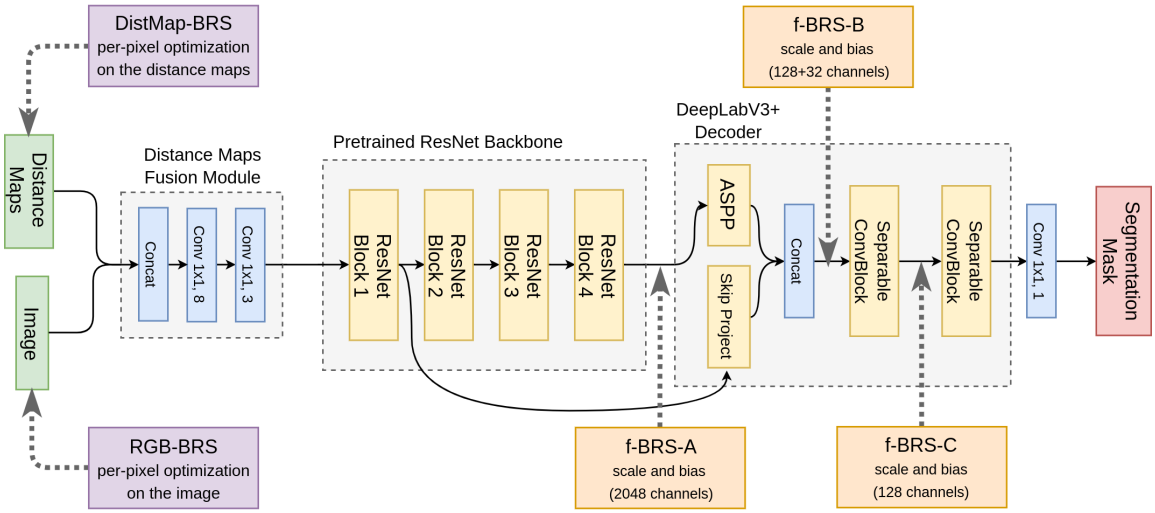

Figure 3.11: Architecture of f-BRS variants.

3.3.2 STM: Space-Time Memory Networks

Memory-based networks are neural networks that maintain an external memory pool where store information. First memory-based networks are proposed for solving NLP tasks such as Q&A, where memorable information is separated into a pair of **key** vector and **value** vector. The **key** vector helps address the **value** vector in the memory. Memory-based models are in charge of mapping the query **key** into an accuracy output **value**. Recently, memory-based models are applied in the Computer Vision field for solving tasks such as long-term visual tracking, movie summarization, and understanding.

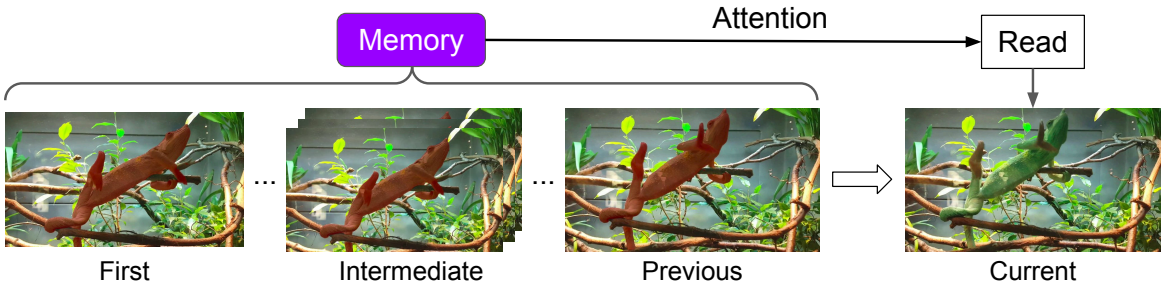

Figure 3.12: Process of STM.

In 2019, Oh proposed STM [19] - a memory-based network for solving semi-supervised video object segmentation task. In the semi-supervised scenario, the mask of the first frame is provided, the user's model has to predict the mask of other frames. Specifically, STM uses an iterative strategy from the second frame to the last frame. Figure 3.12 visualizes how STM works. In frame t^{th}, the mask is produced by extracting **key** and **value** of the current frame. Additionally, the external memory pool is created by concatenating a subset of embedded vectors of previous frames and their masks in the previous steps.

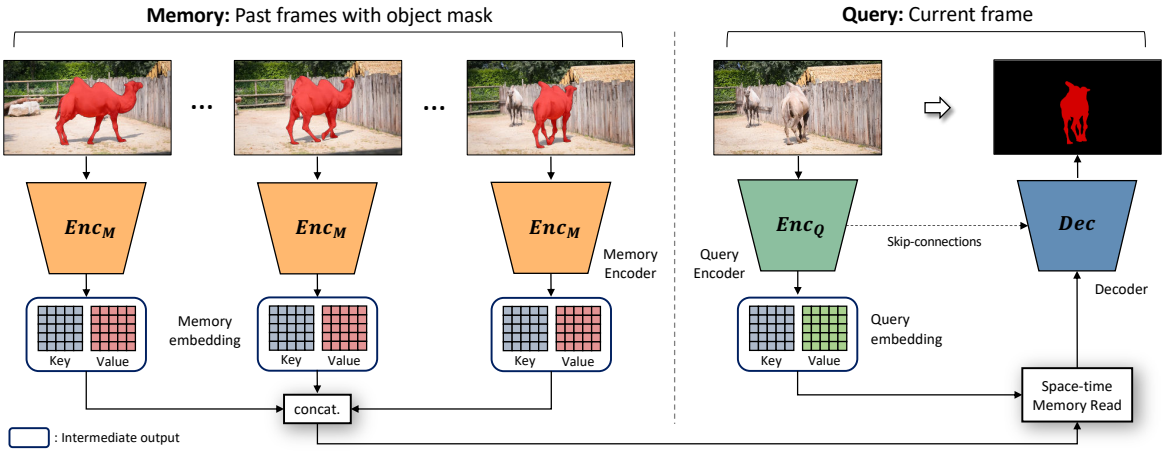

Figure 3.13: Architecture of STM.

Figure 3.13 shows the architecture of STM. Specifically, the current frame is

considered as the query frame, previous frames and their masks are considered as the memory frames. Both the memory frames and the query frame all go through an encoder module to get the **key** and **value** of their frames. **Key** is to address **value**. Depend on the similarity between **key** of the query frames and **key** of the memory frames that model determines which **value** of the memory frames is affected to **value** of the query frame. Therefore, **keys** learn to encode visual features that robust to appearance variations. And **values** learn fine-grained features need to produce the mask. **Keys** and **values** of memory frames are concatenating go through the space-time memory read block along with the **key** and **value** of the query frame. In this block, relative matching scores are computed by the pixel-wise **keys**. Finally, the decoder takes the output of the space-time memory block and reconstruct the mask of the query frame.

Key and Value Embedding: The output of encoder modules is a pair of **key** (dimension size is $H.W.C/8$) and **value** (dimension size is $H.W.C/2$), where H, W, and C is the width, height, and feature dimension of the output in the backbone feature extractor, respectively. In the memory frames, the input of the encoder module is a pair of an RGB frame and its mask. Besides, in the query frame, the input of the encoder module is only an RGB frame. That's the major difference between encoder in the memory frames and the query frame. In case there are more than one memory frame, the **value** of memory frames are stacked along the temporal dimension. Therefore, the dimension of memory frames embedded vectors after stacked are $T.H.C/8$ and $T.H.C/2$, where T is the number of memory frames.

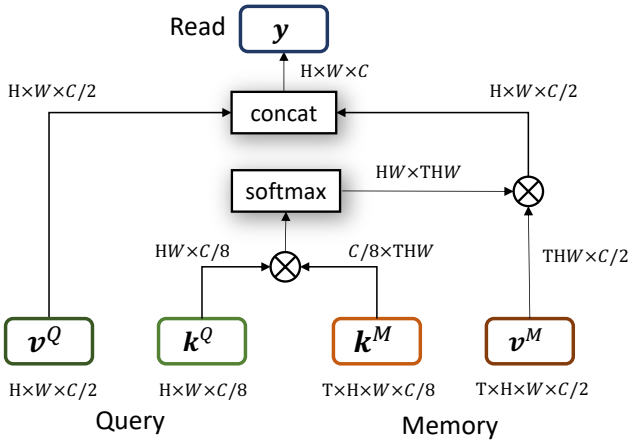

Figure 3.14: Detailed implementation of the space-time memory read.

Space-time Memory-Read: Detailed implementation of space-time memory-read is shown in Figure 3.14. This module is designed to localize the space-time location of the memory. Its implementation is similar to non-local self-attention mechanisms that it performs non-local matching but for attending to other memory frames for retrieval.

Decoder: At the last phase, the output of the read operation goes through an encoder module to reconstruct the object mask of the current frame. The decoder block is built as a stack of convolutional layers and residual blocks. At last, a softmax function is implemented to reconstruct the object mask.

STM tries to memorize more memory frames as many as possible. But there is a limitation of the number of the RGB image and mask pairs of previous frames that STM can memorize because of the limitation of the hardware. Therefore, Oh proposed a strategy for determining what frame is choosing to append to the memory.

CHAPTER 4

PROPOSED METHOD FOR INTERACTIVE VIDEO OBJECT SEGMENTATION

In this chapter, we present our proposed method for interactive video object segmentation. Our proposed method consists of three modules: Control-point-based Scribbles-to-Mask, Self-Reference Refinement, and Guided Mask Propagation. At first, the pipeline of the method is shown. Then, each module is introduced and the detailed implementation is presented.

4.1 Challenges of Interactive Video Object Segmentation

Interactive video object segmentation is a novel task and drawing attention from the research community because of the ability to apply in practice. There are several challenges in interactive video object segmentation that models tackling this task have to solve.

Mask generating from scribbles is hard: For convenience and speedup for the annotator, the inputs in interactive video object segmentation are in friendly annotation form, e.g. scribbles. With the limitation of information from sparse scribbles, models for tackling this challenge have to generate an accuracy mask for all the objects of interest. Depend on the personality and experience of annotator that scribbles have different styles. Figure 4.1 shows the difference between scribble-styles in a video sequence. The masks generating from scribbles also have to achieve semantic meaning.

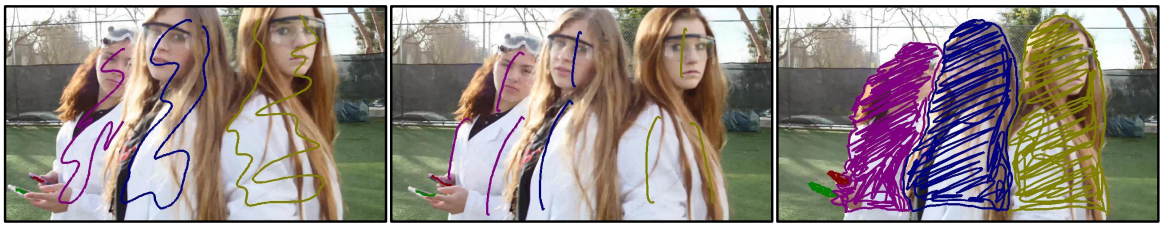

Figure 4.1: Three sets of scribbles come from different annotators in a video sequence.

Accumulated error in mask propagation algorithm: there is the decay from the accumulated error of any propagation algorithm. If the distance between the propagation initial frame and the query frame is too far, there is a chance that the predicted mask is low-quality. The mask propagation algorithm also has to deal with complex multi-object cases, or objects disappear in several frames and reappear. Figure 4.2 shows an instance of complicated multi-object cases on the DAVIS dataset [20]. Therefore, the propagation length has to calculate carefully and has a mechanism to eliminate complex background cases.

Figure 4.2: A video sequence on the DAVIS dataset that has ten objects of interest.

A mechanism that helps the accuracy of the model better after each interaction: For each interaction, the accuracy of the method has to be good enough for reducing the annotator's time need to yield a specific accuracy. Besides, our model is expected to yield better accuracy compared with previous interactions. Therefore, models should have a mechanism for store information

from previous interactions combine with information from the current interaction to generate a better result.

The efficiency of method: speed is a crucial factorial for interactive problems. The annotator cannot wait too long in each round for mask generating and mask propagation. In Section 5.1.2.2, there are metrics about the accumulated time for completing the current interaction. In practice, the mask generating algorithm has to achieve nearly real-time speed.

4.2 Overview of Proposed Method

Our proposed method views scribble as a set of points. At first, a Control-point-based Scribbles-to-Mask module is implemented to yield the mask of the scribbled frame from this set. The propagation phase usually is the bottle-neck on the runtime of video object segmentation methods. By using a memory-based model, with the memory pool store information from previous interactions, our method can be fast, and the accuracy improves by interactions. We use this memory-based model in the Self-Reference Refinement module to refine the mask from the Control-point-based Scribbles-to-Mask module. For improving the quality information fragment of memory pool of memory-based model to have a better reference, we propose a strategy for enriching the memory pool with reliable frames and Multiple Reference Views. At last, a Guided Mask Propagation strategy is proposed to eliminate complex backgrounds for performance improvement.

4.3 Proposed Method for Interactive Video Object Segmentation

4.3.1 Pipeline

Figure 4.3 illustrates the pipeline of our proposed method.

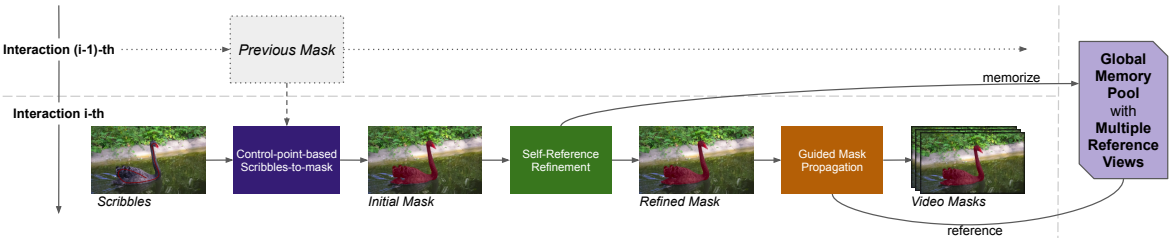

Figure 4.3: The pipeline of our proposed method.

For each interaction, our proposed method consists of two stages: image segmentation mask generating and video object mask propagation.

In the first stage, a Control-point-based Scribbles-to-Mask module that takes input as scribble paths and outputs an initial mask is proposed. This initial mask will further refined by the Self-Reference Refinement module, which is based on a memory-based model, to yield better accuracy.

In the second stage, the refined mask is propagated to neighbor frames using a memory-based model. This memory-based model maintains a memory pool, called Global Memory Pool. This memory pool is filled with all pairs of frames having scribbles in current and previous interactions combine with their masks. Furthermore, we propose to enrich the Global Memory Pool with the Reliably Inferred Frames, obtained in the propagation process. In this way, we have more high-quality referenced views for the objects of interest. With the approach, all frames is only influenced by its nearest reliable frame. A Guided Mask Propagation strategy is also proposed to help the model focus on potential regions of interest in a frame, as suggested in [29].

4.3.2 Control-point-based Scribbles-to-Mask

Figure 4.4 illustrates the Scribbles-to-Mask module.

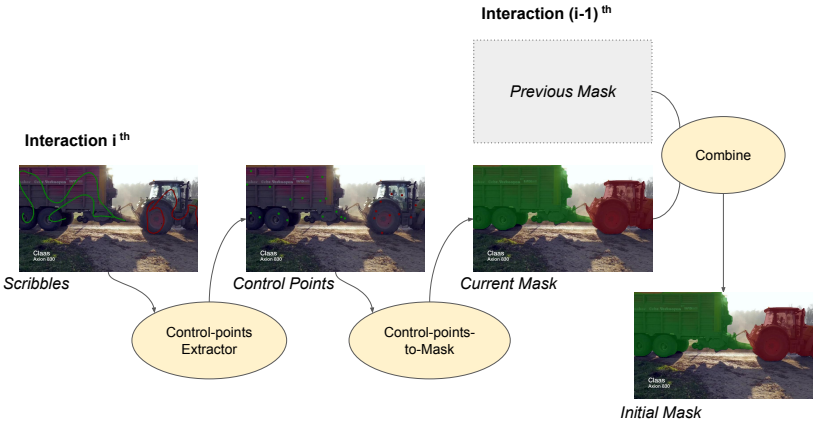

Figure 4.4: Overview of the Scribbles-to-Mask module.

In the first interaction, all scribbles are converted into control points by a control point extractor. However, in the subsequent interactions, scribbles only point out falsely predicted regions. To guarantee that all control points are in the center zone of segment regions, we separated scribbles into two groups: boundary scribbles, and inside scribbles. Boundary scribbles are scribbles that fix the boundary object and they are minor fixes. Inside scribbles are scribbles that point out falsely predicted large regions of the objects and they are major fixes. Boundary scribbles with be dilated and overwrite the previous mask. Inside scribbles will be converted into control points and used in control-point based mask generating. We separated two groups of scribbles by dilating the boundary in the mask of all the objects. Scribbles that have an enormous overlap length with dilated boundary mask are boundary scribbles. Remaining scribbles are inside scribbles. Figure 4.5 visualizes how to generate this dilated boundary mask. First, with each object, our method finds the contour boundary of the mask of each object and dilates it with a fixed size. At last, we will merge the dilated mask of objects into a multi-object dilated boundary mask.

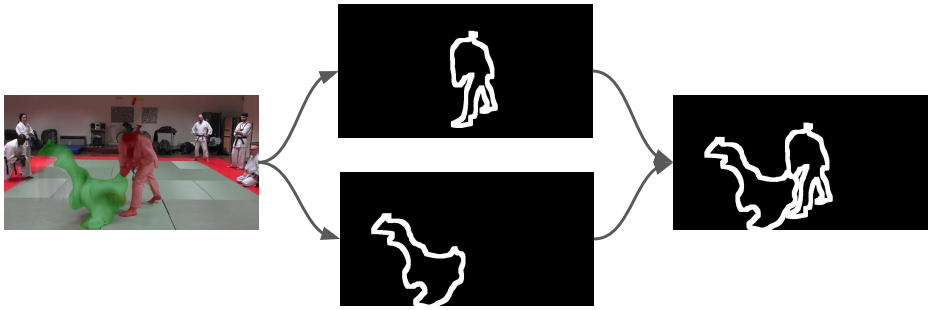

Figure 4.5: Boundary of object regions are dilated and merged.

4.3.2.1 Control-points Extractor

Scribbles are sets of points. At first, our method converts scribbles into control points. There is a trade-off between accuracy and speed by changing the number of generated control points. Figure 4.6 shows examples of the Scribbles-to-Mask module output with a certain number of control points. In our work, we set a maximum number of control points that scribbles are converted into. By this approach, we can control the maximum time that the Scribbles-to-Mask module with spend. The number of control points that each scribble is converted into is the same as others. In each scribble, control points will be evenly sampled along with the scribble. Each of these control points holds the information about its coordinates on the image and the object to which it belongs.

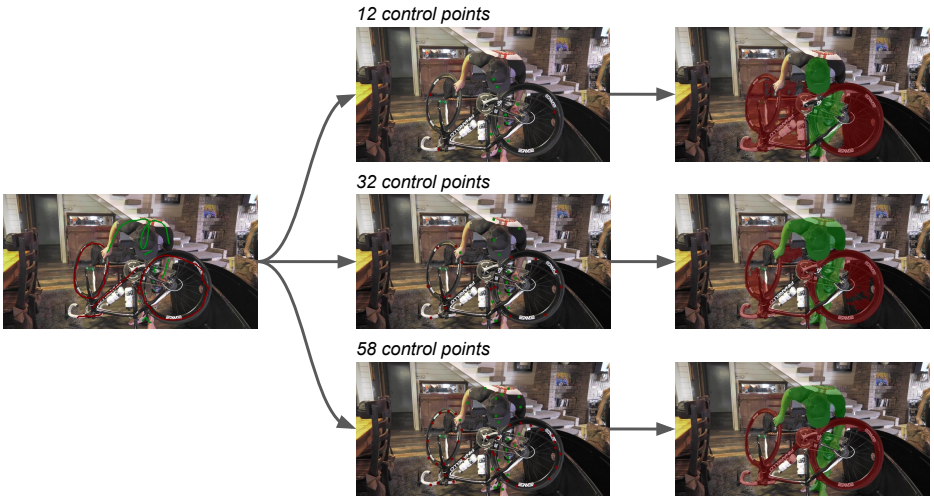

Figure 4.6: The difference between the predicted mask in the Scribbles-to-Mask module from several control points configurations.

4.3.2.2 Control-points-to-Mask

To generate an individual object mask from the image, an encoder-decoder model takes as input two sets of points, one containing positive points: points that belong to the object, one containing the negative points: points that do not belong to the object. We consider control points belonging to that object positive points and control points belonging to other objects, often including the background class, as negative points. This model will generate a probability map for each object. The map will be further be refined by a backpropagating refinement algorithm, called f-BRS as staged in Section 3.3.1. In the final step, the multi-object segmentation mask is generated by merging the probability map of objects. Each pixel in the multi-object mask saves the object that its probability is the largest between the objects of interest. Figure 4.7 shows an example of the Control-points-to-Mask module.

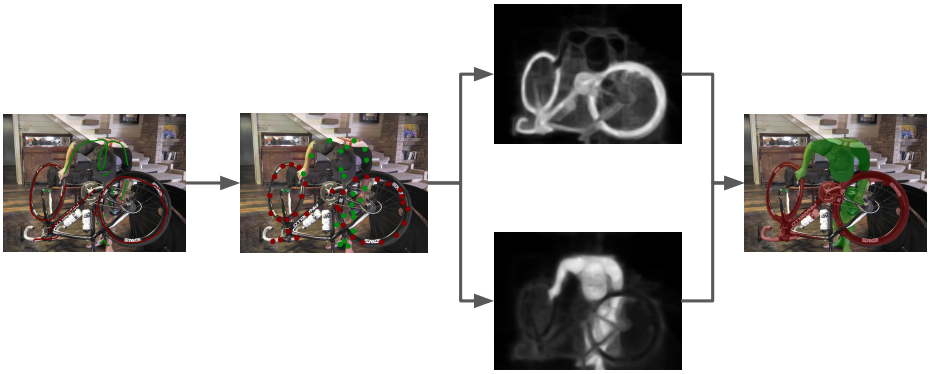

Figure 4.7: An example of the Control-points-to-Mask module.

4.3.3 Global Memory Pool

The core algorithm used in the remaining components is based on a memory-based video object segmentation model. This memory-based model maintains a memory pool that can be filled with pairs of frames and their corresponding masks. They perform segmentation on the query frame with reference to that memory pool. We use such models in two different ways: as a refinement mechanism in the Self-Reference Refinement module and as a mask propagator in the Guided Mask Propagation module. In our work, we utilize the Space-Time Memory Networks [19] as the memory-based model.

For each sequence, our method maintains a memory pool that is global across interactions, called Global Memory Pool.

Naturally, our method relies heavily on the quality of the memory fragments stored in the Global Memory Pool. The most important fragments are the scribbled frames obtained after each interaction. Another set of frames to consider are the Reliably Inferred Frames: the predicted masks that are considered reliable obtained throughout the propagation process. There is a limitation of hardware so Global Memory Pool cannot store all pairs of frames and their mask. Therefore, we propose and experiment with choosing Reliably Inferred Frames

strategies, the detailed is in Section 5.2.1.

4.3.4 Self-Reference Refinement

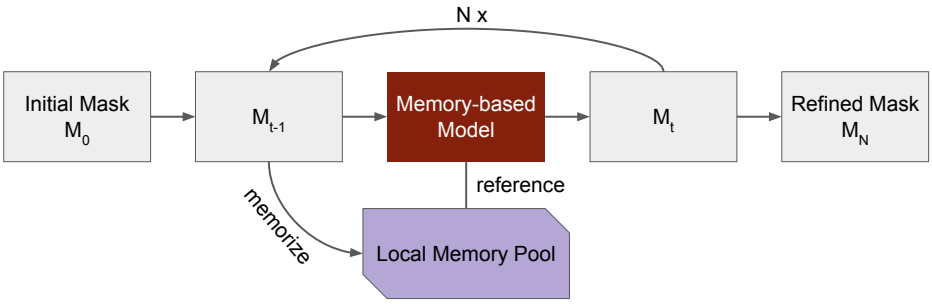

Figure 4.8: The Self-Reference Refinement module.

Figure 4.8 illustrates the Self-Reference Refinement module.

As a refinement mechanism, the memory-based model maintains a Local Memory Pool that memorizes all previously inferred pairs of frame and its mask. First, our model stores the pair of the scribbled frame and its mask in the memory pool. Second, our method uses that memory pool to perform segmentation on the scribbled frame. This procedure repeats a specific number of iterations. In our work, we utilize Local Memory Pool with the aforementioned Global Memory Pool. The refined mask brings more information about previous interactions compare with the initial mask.

Figure 4.9 shows examples where the Self-Reference Refinement module helps to refine the initial mask. The example on the left shows how it can help with under-segmentation cases, specifically better boundary snapping and hole filling. The second example shows how it can also help with over-segmentation cases, specifically with long thin objects, albeit not perfectly.

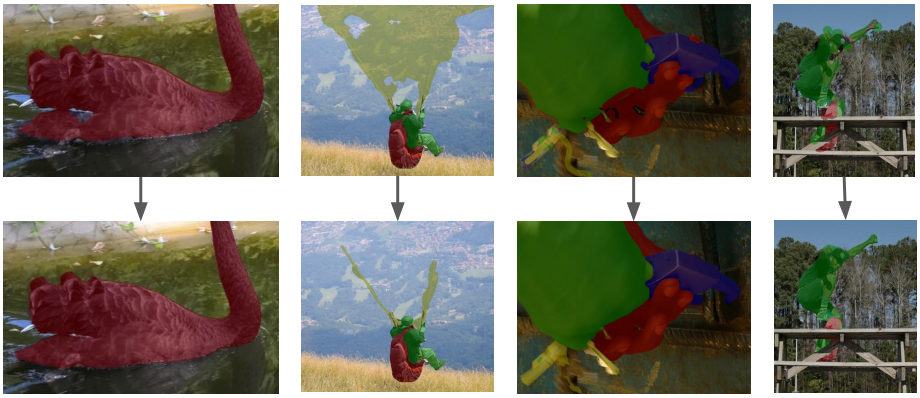

Figure 4.9: Cases that the Self-Reference Refinement module helps improve the result: under-segmentation, over-segmentation.

4.3.5 Guided Mask Propagation

In each interaction, with the current annotated frame as the initiator, the propagation goes in two directions to both ends of the video sequence. However, the process does not go all the way but instead at length calculated by considering the indices of the annotated frames in previous interactions. As can be seen from Figure 4.10, it stops midway between the current annotated frame index and the nearest previous annotated frame index in both directions.

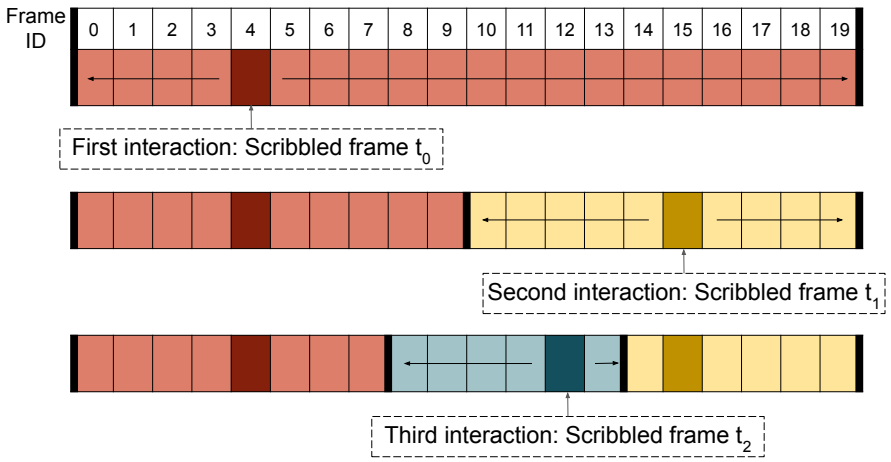

Figure 4.10: All frames are only influenced by its closest reliable frame.

Specifically, in the interaction i, given f_i is the annotated frame index, l_i is the left-most index of propagation frames, and r_i is the right-most index of propagation frames.

Then, l_i value is computed by:

$$l_i = \begin{cases} 0, & \text{if } i = 1 \text{ or } \forall j < i, f_j > f_i \\ \left[(f_i + \max_{j<i, f_j \leq f_i} f_j)/2 \right], & \text{otherwise} \end{cases}$$

Similarly, r_i value is computed by:

$$r_i = \begin{cases} n_{frames} - 1, & \text{if } i = 1 \text{ or } \forall j < i, f_j < f_i \\ \left[(f_i + \min_{j<i, f_j \geq f_i} f_j)/2 \right], & \text{otherwise} \end{cases}$$

Where n_{frames} is the number of frames in the video sequence.

Considering the possible decay from the accumulated error of any propagation algorithm, this is a reasonable approach as each frame is only influenced by its nearest reliable frame.

To counter the problem of misidentification or over-segmentation, we employ a guidance mechanism in our propagation module. It uses the predicted masks of neighboring frames to generate a localization mask. We use this mask to filter out potentially unrelated surroundings so that the segmentation algorithm can have a better focus on what to segment. Figure 4.11 shows how the Guided Mask Propagation module focuses on regions of interest in the query frame t^{th} by the predicted mask of frame $(t-1)^{th}$. First, the multi-object mask of the frame $(t-1)^{th}$ is converted into a binary mask that points out object regions. Next, this binary mask will be dilated and blurred with specific kernel sizes. At last, the module computed the convolution between this blurred mask and the

frame t^{th}. This convoluted image is using for query instead of the original frame t^{th}.

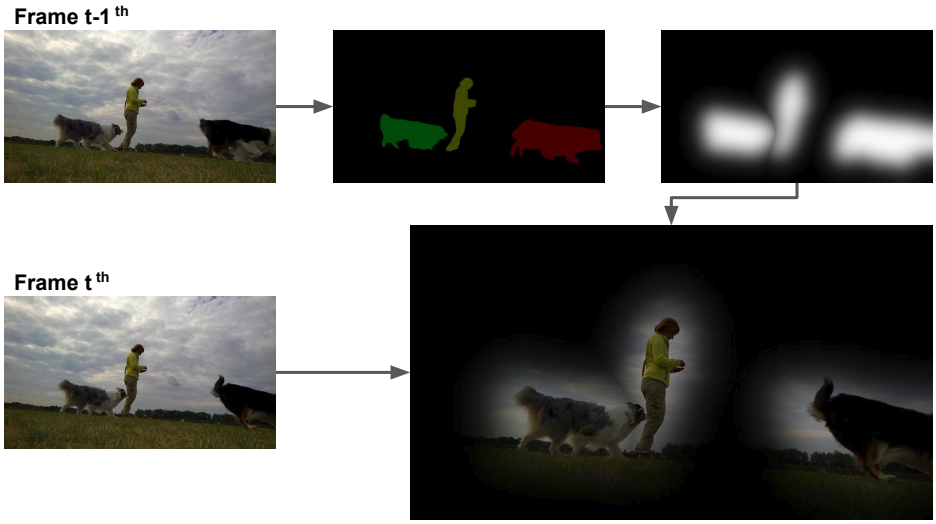

Figure 4.11: Guidance mechanism for the query frame t^{th}.

CHAPTER 5
EXPERIMENTAL RESULTS

In this chapter, we describe the dataset, evaluation platform, and evaluation metrics. Besides, we experiment with different configurations of our proposed method. With that, we also analyze the result and report observations. At last, the result of our proposed method in the Interactive Scenario of The DAVIS Challenge 2020 is shown.

5.1 DAVIS Framework For Interactive Video Object Segmentation

5.1.1 Densely Annotated VIdeo Segmentation (DAVIS) Dataset

The DAVIS 2017 dataset [20] is a public dataset that focuses on multiple-objects video segmentation.

There are four sets of video sequences: `Train`, `Val`, `Test-Dev`, and `Test-Challenge`. Video sequences are lists of RGB frame images. The length of a video sequence is around 70 frames. Most of video sequences are in 4K resolution, but there are also some 1440p, 1080p, 720p images. In `Train` set and `Val` set, annotation masks of all frames in videos are provided. Figure 5.1 shows examples of image's annotations in the DAVIS 2017 dataset.

Figure 5.1: Example annotations of the DAVIS 2017 dataset.

In the DAVIS 2017 dataset, objects are considered as moving items in the video sequence. Objects are split by their semantics, although they might have the same motion orbit. Specifically, with people and animals included their clothes (including helmet, shoes, cap, etc.) are considered as a single instance. Items that are carried and easily separated (such as bags, skies, poles) are considered as other objects.

Compare with previous datasets, the DAVIS 2017 dataset provides a new standard about the quality of video resolutions and annotations. Video sequences of DAVIS also have multiple complicated scenarios such as multi-object interactions, camera motion, and occasions.

The DAVIS 2017 dataset has moved to maintenance mode. Therefore, The DAVIS 2020 Challenge will be the last time that The DAVIS Challenge is organized.

5.1.2 DAVIS Interactive Evaluation Framework

5.1.2.1 Overview

The DAVIS Interactive Evaluation Framework is designed for evaluation performance of models in the DAVIS Interactive track [30].

There are two types of connection for communication between the user and the evaluation server: local and remote. Local run in the `Val` set, is used for method validation. Ground truth masks of frames in `Val` set are provided so there will be no need to connected the organizer's server for evaluation. Remote is run in the the `Test-Dev` set and require to connect to the DAVIS organizer's server, the result of remote will be counted as the result for the challenge. At the beginning of a connection, the framework will generate a unique *session_id* for session management and logging.

Human-annotated scribbles: For each sequence in the `Train`, `Val`, and `Test-Dev` set, three scribbles sets in three particular frames are annotated by different annotators. Three scribbles set of a sequence is shown in Figure 5.2. One of these scribbles sets will be provided in the first interaction. With different scribbles sets for each interaction, the user model has to produce a good result with all annotator perspective, which will need in realistic applications.

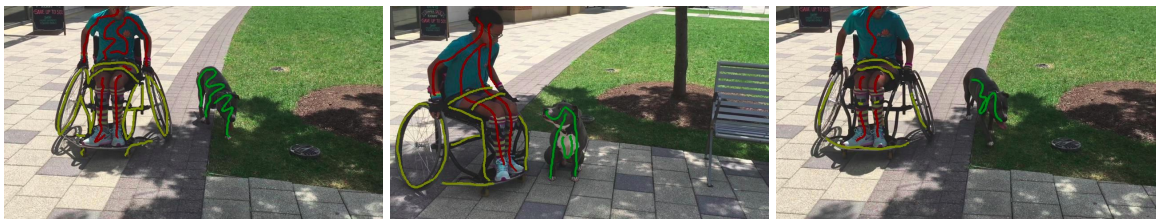

Figure 5.2: Three human-annotated scribbles set for sequence "girl-dog".

Human-simulated scribbles: During subsequent interactions, from segmen-

tation masks submitted for the user, the server chooses the worst result mask and annotates this mask, a robot will point out false prediction regions in the mask by scribbles, as a human would do. Figure 5.3 shows an example of a human-simulated scribbles set.

Figure 5.3: An instance of human-simulated scribbles set in the subsequent interactions.

The framework also supports other tasks such as visualizing frame image with overlay mask, merging instance probability maps into one multi-object mask.

5.1.2.2 Evaluation Metrics

There is a trade-off between the accuracy and speed in interactive video object segmentation task. AUC and $\mathcal{J}\&\mathcal{F}$@60s [30] are used for evaluating models that tackle this task by leverage efficiency and effectiveness.

$\mathcal{J}\&\mathcal{F}$ [31] is a well-known metric and widely used for evaluating the accuracy in the video object segmentation task. $\mathcal{J}\&\mathcal{F}$ is defined by the average of the region similarity $\mathcal{J}$ and the contour accuracy $\mathcal{F}$.

- Region Similarity ($\mathcal{J}$): Since its invention, the Jaccard index, which is also known as intersection-over-union, is a common metric to compute the pixel-level similarity between the ground truth mask G and the predicted mask

M. The Jaccard index is defined as:

$$\mathcal{J} = \frac{|M \cap G|}{|M \cup G|} = \frac{|M \cap G|}{|M| + |G| - |M \cap G|}$$

- Contour Accuracy ($\mathcal{F}$): To compute the similarity between ground truth mask G and the predicted mask M based on contours, the contour accuracy $\mathcal{F}$ is proposed. Given $c(G)$ and $c(M)$ are the set of contour points in G and M, respectively. A bipartite graph matching is applied to map between contour points in $c(G)$ and $c(M)$. From that, the precision P_c and the recall R_c are computed between two sets. The contour accuracy F is defined as:

$$\mathcal{F} = \frac{2P_cR_c}{P_c + R_c}$$

For evaluating the speed of methods in interactive video object segmentation, the time that needs for each interaction is saved. After each interaction, the server computes the $\mathcal{J}\&\mathcal{F}$ score and the accumulated time from the start of the first interaction to the end of the current interaction. A curve showing the accuracy $\mathcal{J}\&\mathcal{F}$ with accumulated time needs to achieve that accuracy is plotted. An instance of these curves is shown in Figure 5.4. From the curve, two metrics are computed for evaluating and ranking between models:

- **AUC**: Shorted for the Area Under The Curve. The area of the region under the curve is computed and normalized by the total time.
- $\mathcal{J}\&\mathcal{F}$**@60s**: An interpolation of the curve is used for evaluating the accuracy at exactly 60 seconds from the start of the first interaction. This metric encourages users to optimize the speed of models to achieve good accuracy in a short time.

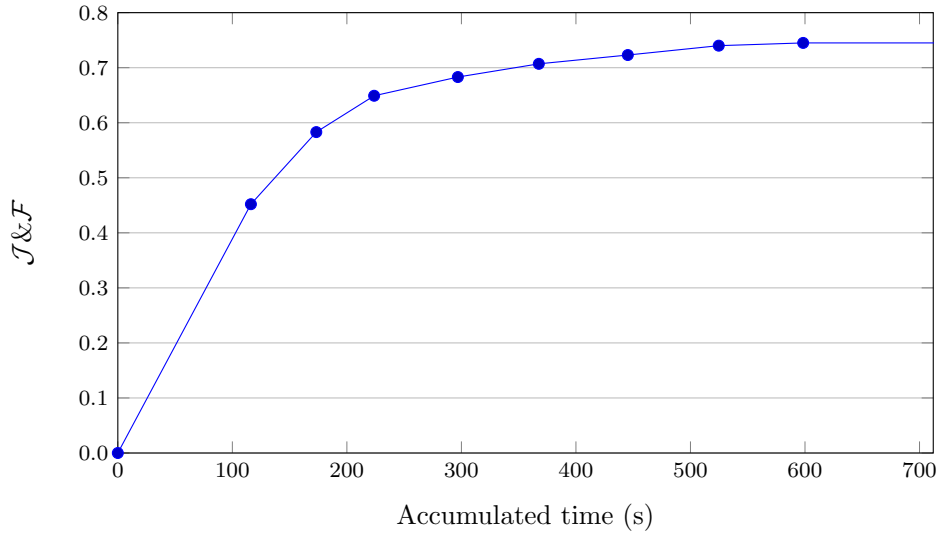

Figure 5.4: A curve showing the accuracy with accumulated time.

5.2 Experiments

5.2.1 Experimental Setup

As mentioned in Section 4.3, our method comprises two stages: scribble-based image segmentation mask generation and mask propagation. For improving the accuracy in each stage, refinement, and guidance modules are proposed and leverage. In this work, we mainly experiment in generating control points strategy, guidance information for the query frame, what frames used for reference, and choosing propagation range.

Generating Control Points

There is a trade-off between the accuracy and the speed by changing the number of control points that scribbles are converted into. In the DAVIS Challenge, we set 50 is this maximum number. In the subsequent interactions of our method, when the annotator tries to correct the image segmentation mask, the scribbles are divided into two groups: boundary scribbles, and inside scribbles. The detail is staged in Section 4.3.2.1. We experiment with the strategy for separating scrib-

bles into these two groups. At first, from the mask of the previous interaction, the boundaries of objects are dilated with kernel size is 15. Figure 5.5 shows an instance of these dilated boundary masks. In the first experiment scenario, these annotation scribbles that have a 10% length overlap with the dilated boundary mask are considered as boundary scribbles and other scribbles are considered as inside scribbles. In the second experiment, scribbles are split into two smaller scribbles: the part that overlaps with the image consider as boundary scribbles, the other part consider as inside scribbles.

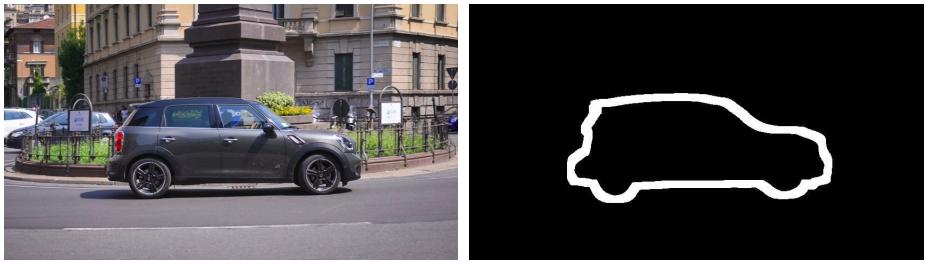

Figure 5.5: Left: the frame image. Right: the dilated boundary mask with kernel size is 15.

Guidance Information For The Query Frame

For helping the memory-based model focus on the potential region of objects in the query image, a guided strategy is implemented, as Tran et al. suggested [29]. Figure 5.6 shows examples of query frames after applied guidance.

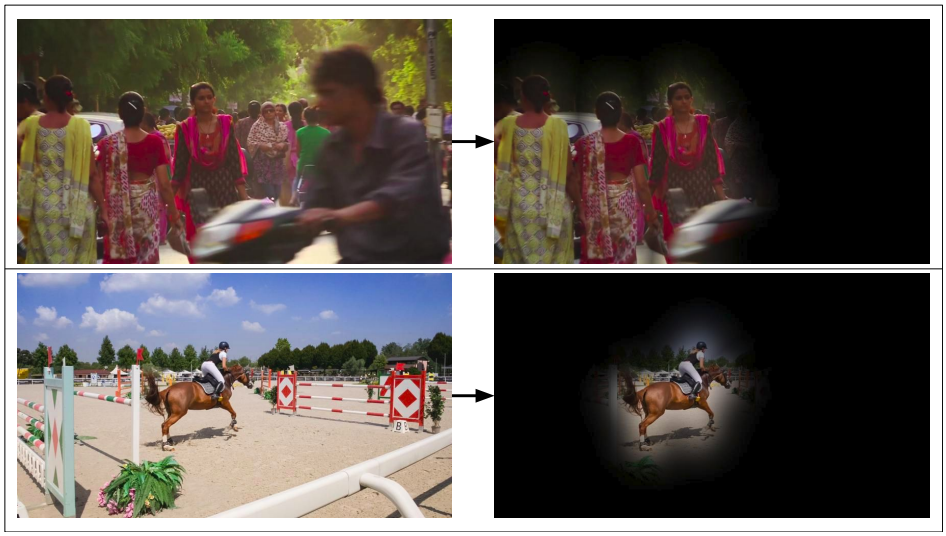

Figure 5.6: Examples of query frames before and after applied guidance information.

In our work, we fixed the dilated size is 20 pixels and the Gaussian blur size is 30 pixels. We experiment and compare the AUC and $\mathcal{J}\&\mathcal{F}$@60s score of our model when without guided, guided for the query frames only, and guided for the query frames and reference frames.

Propagation Range And Choosing Reference Frames Strategy

Because of the limitation of GPU memory, the maximum number of frames for reference that the memory-based model can be stored is 15. Besides the annotated frame in previous interactions, our model also tries to memorize intermediate frames in the propagation process. The quality of reference frames is important for the accuracy of the model. The more high rich information of reference and more relevant between the reference frames and the query frame, the more accuracy that the query image can achieve. The propagation stage usually is the bottle-neck in the method's speed. Besides, any propagation algorithm has accumulated error. Therefore, the propagation range should not be too broad but still need to cover a set of neighbor relevant frames. At first, we

experiment with the propagation strategy that after each interaction, the query frames are propagated from the nearest scribbled frame from previous interactions, as staged in Section 4.3. With that approach, in case too many interactions are performed, there is a chance that the propagation range in last interactions is tiny, even it can equal to 1. Therefore, we also experiment with the minimum propagation for each direction of propagation. In our work, a constant equal to 10 is set. In experiments with reference frames, besides annotated frames always in the memory pool, we suggest that the intermediate frames are evenly sampled with the size depending on the remaining size of the memory pool. Another way is only used the nearest previous frames of intermediate frames for reference.

5.2.2 Experimental Results

Experiments are performed by changing the configuration in each module:

- **Configuration 1:** the first scenario in generating control points is implemented; guidance information is not used; there is no minimum length of propagation range.
- **Configuration 2:** the second scenario in generating control points is implemented; guidance information is not used; there is no minimum length of propagation range.
- **Configuration 3:** the first scenario in generating control points is implemented; guidance information is used for the query frame only; there is no minimum length of propagation range.
- **Configuration 4:** the first scenario in generating control points is implemented; guidance information is used for the query frame and reference frames; there is no minimum length of propagation range, all annotated frames is stored in the memory pool and reference intermediate frames are

evenly sampled.

- **Configuration 5:** the first scenario in generating control points is implemented; guidance information is not used; the minimum length of propagation range is 10, all annotated frames is stored in the memory pool and reference intermediate frames are evenly sampled.
- **Configuration 6:** the first scenario in generating control points is implemented; guidance information is not used; the minimum length of propagation range is 10, only annotated frames that in the propagation frames are stored in the memory pool and reference intermediate frames are evenly sampled.
- **Configuration 7:** the first scenario in generating control points is implemented; guidance information is not used; the minimum length of propagation range is 10, only annotated frames that in the propagation frames are stored in the memory pool and the nearest previous intermediate frames are used for reference.

The quantitative result of configurations are shown in Table 5.1

Table 5.1: Quantitative result of 7 configurations on the DAVIS 2017 `Val` dataset

Configurations	**AUC**	$\mathcal{J}\&\mathcal{F}$**@60s**
Configuration 1	**0.816**	**0.824**
Configuration 2	0.803	0.821
Configuration 3	0.751	0.768
Configuration 4	0.758	0.768
Configuration 5	0.808	0.820
Configuration 6	0.805	0.818
Configuration 7	0.813	0.821

From the result of configurations, we can yield some observations. The first scenario in generating control points proves its accuracy compare with the second scenario. It means that in the mask correcting step, our method still needs to be improved boundary of the object's segmentation region, and a limitation of the number of inside scribbles is enough to fix previous heavily wrong segmentation regions such as re-id false prediction. Depend on the speed of the objects in video sequences that the size of the guided kernels is carefully chosen. Besides, a more annotated frame in the memory pool corresponds with the more information that the query frame can yield, therefore in each interaction, all previous annotated frames should be stored in the memory pool if it is possible.

Experiments are performed on two NVIDIA GTX 2080 Ti GPUs. It takes 4.31 seconds on average per object in the mask generation phase and 0.21 seconds on average per frame in the mask propagation phase.

5.2.2.1 Qualitative Results

The best setting using for evaluation is **Configuration 1**. Qualitative results of our proposed method are visualized in Figure 5.7. After the last interaction, our proposed method can easily handle complicated multiple-object cases.

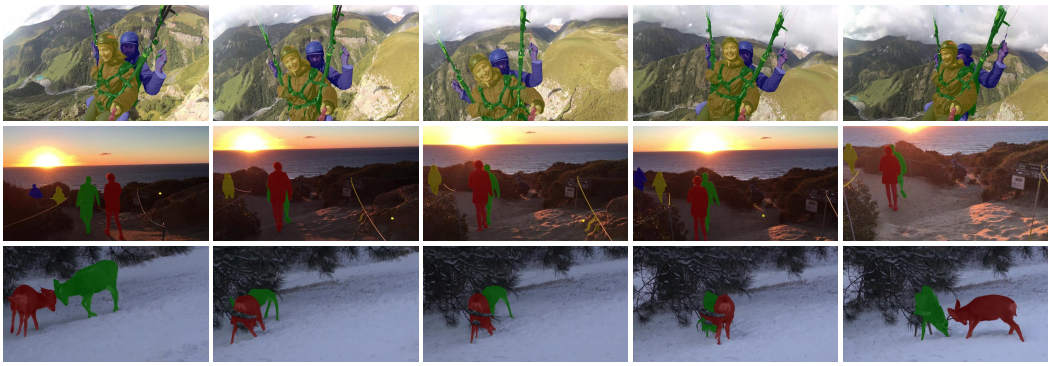

Figure 5.7: Final masks after the last interaction of three video sequences.

5.2.2.2 Result in The DAVIS Challenge 2020

As shown in Table 5.2, our method achieves 0.767 and 0.759 in terms of Area Under the Curve (AUC) and $\mathcal{J}\&\mathcal{F}$ at 60 seconds ($\mathcal{J}\&\mathcal{F}$@60s), respectively, ranking 2^{nd} in Interactive Scenario of The DAVIS Challenge 2020. Figure 5.8 visualizes our models accuracy with accumulated time.

Table 5.2: Quantitative result in The DAVIS 2020 Challenge.

Position	Participant	AUC	$\mathcal{J}\&\mathcal{F}$@60s
1	Heo et al. [ECCV 2020] [32]	0.772	0.787
2	**Ours**	**0.767**	**0.759**
3	Liang et al. [CVPR 2020] [25]	0.754	0.762
4	Nguyen et al.	0.468	0.473

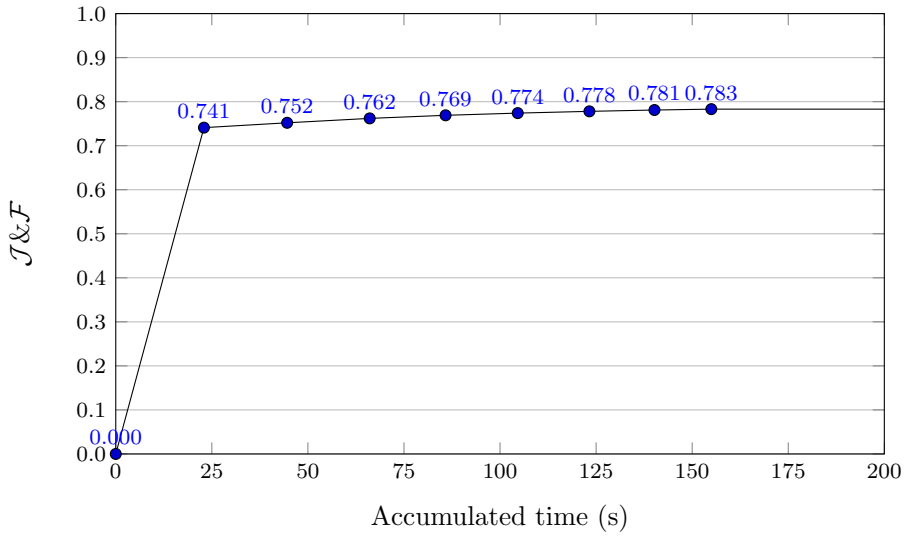

Figure 5.8: The AUC plot of our proposed method (In first 200 seconds).

Note: In The DAVIS Challenge 2020, the main metrics for ranking is AUC score.

CHAPTER 6

APPLICATIONS

In this chapter, we present the applications of our proposed method for interactive video object segmentation, including a video object segmentation annotation tool and a semantic video website. The layout and features of each application are shown in this chapter.

6.1 Video Object Segmentation Annotation Tool

6.1.1 Overview

Figure 6.1 shows the layout of our annotation tool.

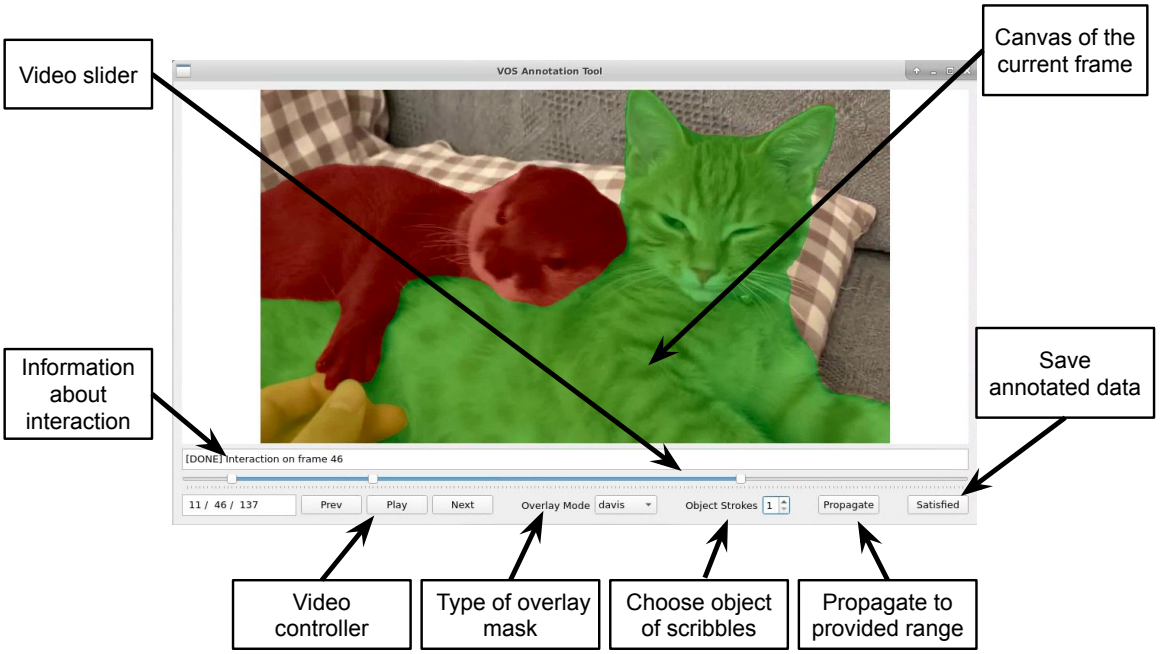

Figure 6.1: Layout of the video object segmentation annotation tool.

It is designed in the Qt platform with core algorithms are from our proposed method for interactive video object segmentation.

6.1.2 Main Features

Play video and choose the annotated frame: Video will be played when annotators click the "Play" button. Annotators can choose a arbitrary frame by dragging the slider center handle or translating frame by "Next" and "Prev" buttons. They also can play video and pause at the desired frame. The propagation range for each interaction is supported by dragging the left-most and right-most handle. There is a text box for showing indices of the left-most frame of propagation range, the current frame, and the right-most frame of propagation range, respectively. Figure 6.2 shows components that support this feature.

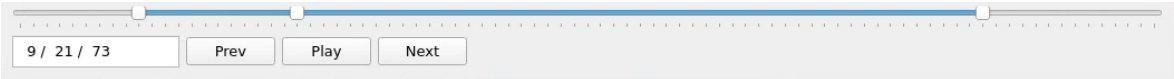

Figure 6.2: Components that support play video and choose the annotated frame.

Generate image segmentation mask from scribbles: Annotators can draw scribbles in the canvas over the frame. Scribbles have their color that corresponds with the color map using for the DAVIS dataset [20]. The color of background-scribbles is black. After the annotator releases pressing event when completing drawing a scribble, our tool generates a new multi-object mask in nearly real-time speed. The annotator can refine this mask by drawing scribble repeatedly. Figure 6.3 shows the image segmentation mask after appending new scribbles.

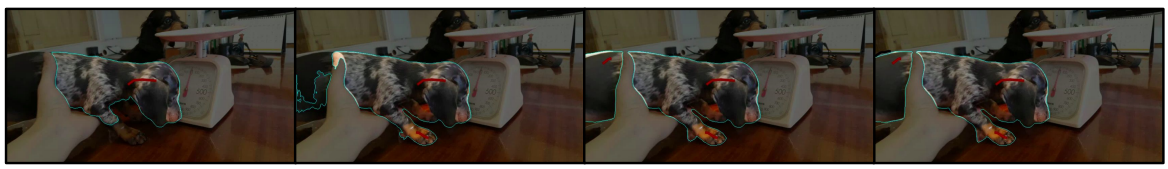

Figure 6.3: The image segmentation mask of a frame after appending new scribbles.

Choose the type of overlay mask: There are four types of overlay mask that we support in this tool such as **fade**, **DAVIS**, **checker background**, and **solid background**. In **fade**, we highlight the boundary of object regions and reduce the brightness of the background region. In **DAVIS**, object regions have their color that corresponds with the color map using for the DAVIS dataset [20]. In **checker background**, the background region is replaced with the checker pattern. In **solid background**, the background region is replaced with a solid color. Each type has different advantages and disadvantages depending on the use-case and the experience of the annotator. Figure 6.4 visualizes the view of each type of overlay mask.

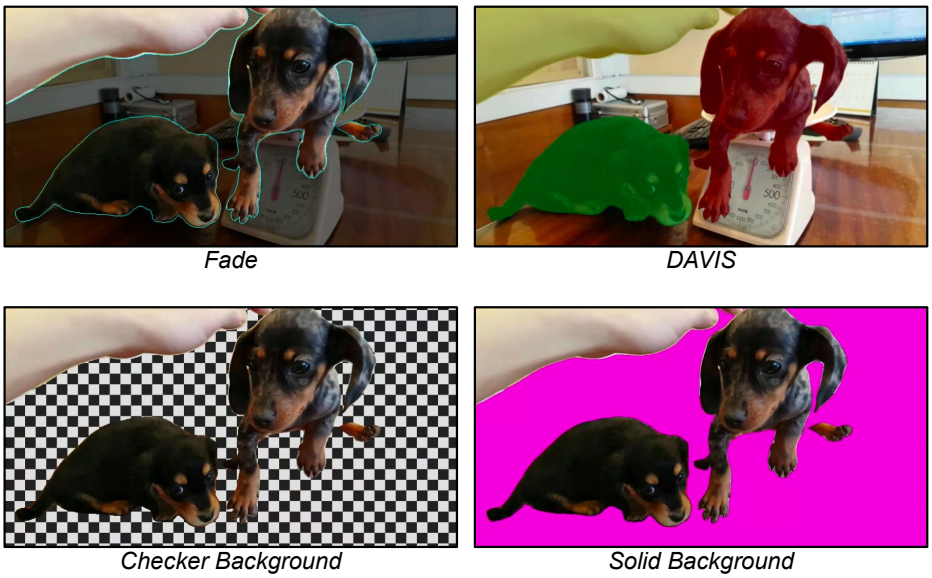

Figure 6.4: Types of overlay mask.

Propagate mask with provided range: When the annotator is satisfied with the image segmentation mask of the current frame and wants to propagate to neighbor frames. Our tool only propagates in the range that the user provided. It speeds up the propagation stage and maintains the quality of the mask in frames that far from the propagation initial frame. It is recommended that each

propagation range is a consistent scene without switching.

Save the annotated data and visualize in the Semantic Video website: When the annotator is satisfied with the segmentation mask of all frames in the video and clicks the "Satisfied" button, out tool will store the annotated data into a database. Besides, it also creates a new annotation session in the Semantic Video website for visualization and interaction.

6.1.3 Annotation Flow

Figure 6.5 illustrates the annotation flow for a video.

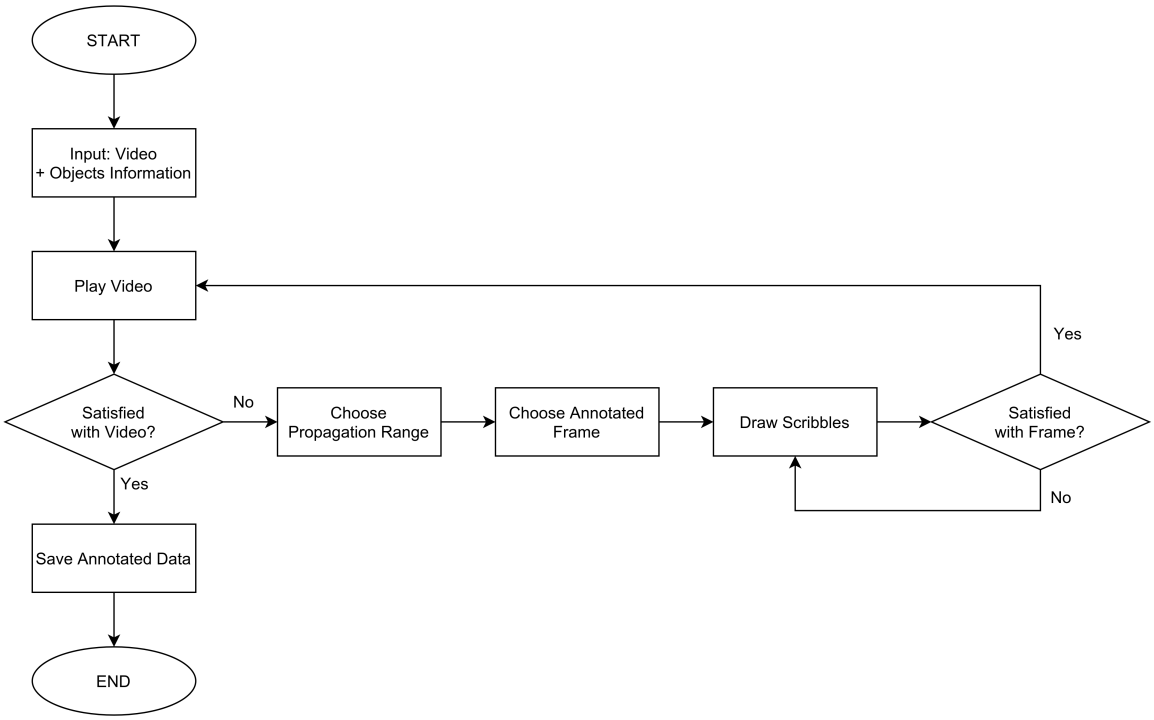

Figure 6.5: The annotation flow for a video sequence.

At first, the annotator plays the video frame by frame. Next, the annotator chooses a propagation range and a high-information frame for annotation. In the annotated frame, the annotator provides scribbles by choosing the object index

and draw the scribbles path. The tool will create a new image segmentation mask after each interaction. Until satisfied with the segmentation mask, the annotator refines this mask by drawing scribbles repeatedly. He/she also can choose different types of overlay mask to see small mistakes in the mask. After satisfied with the current mask, the annotator presses the "Propagate" button to neighbor frames with the provided range.

Again, the annotator plays video and chooses a propagation range and a frame currently does not segment correctly for drawing scribbles in the mismatch regions. The loop continues until the annotator is satisfied with the quality of the segmentation mask of all the frames.

The annotation result is shown in the Semantic Video website, the detail is in Section 6.2.

6.1.4 Configuration

- GPU usage: what GPU is used for the annotation process.
- Memory size: The size of the memory pool of memory-based model used in the propagation stage.
- Objects: Number of objects, and their metadata.
- Step frames: In case the length of the annotated video is too long for annotated all frames. We proposed a strategy that the annotator just has to annotate pivot frames. The distance between two consecutive pivot frames indices is specified by the step frames.

6.2 Semantic Video Website

The layout of our Semantic Video website is shown in Figure 6.6.

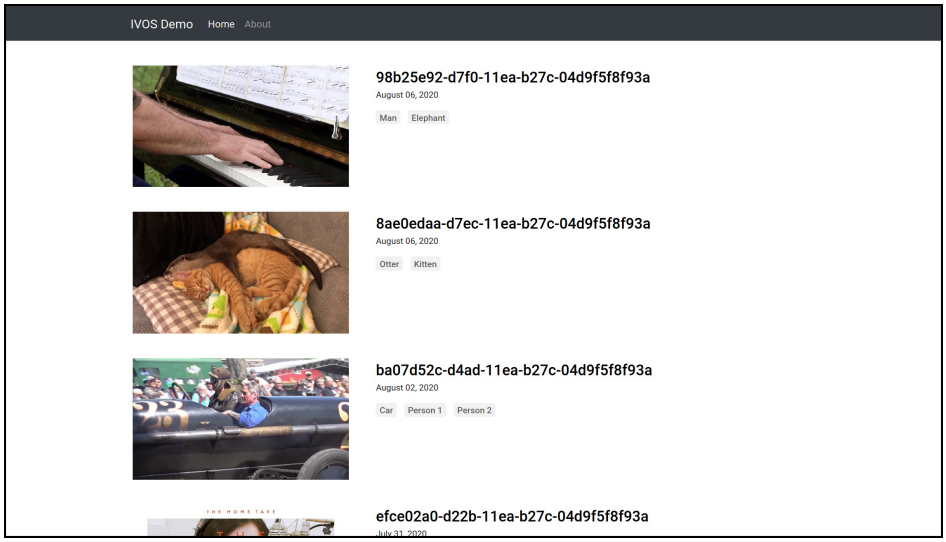

Figure 6.6: Layout of the Semantic Video website.

It shows a list of video's annotation sessions. In each annotation session, our website displays the annotation session name, the annotation created date, and the name of objects.

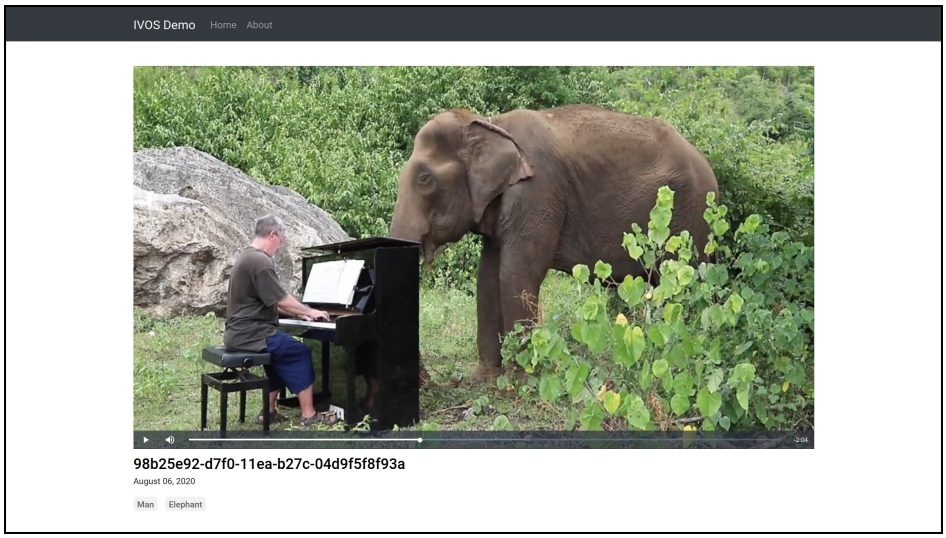

Figure 6.7: Layout of a session page.

Figure 6.7 shows an annotation session page. Besides information about the ses-

sion, our website also renders a video player that plays the annotated video. First, the user plays the video and pause at a arbitrary frame. The server computes the segmentation mask and returns it to the client. As shown in Figure 6.8, when the user hovers above the region of an object, all segmentation regions of that object will be highlighted. When the user clicks the region of an object, predefined metadata of that object will be shown. One of its applications is in education when our website can help kids learn about the shape of each object and show object's media information such as picture, video, music, and sound.

Figure 6.8: Segmentation masks display in the video player when the user hovers object regions.

The segmentation mask of an object in a frame is computed as follows: if the frame is one of the pivot ones, the server just simply returns the stored mask of the pivot frame to the client, otherwise, it is computed by a memory-based model with reference to nearest pivot frames. This strategy is called **lazy propagation**. It provides the ability to handle videos of arbitrary length.

The website is served by Flask with Gunicorn for the WSGI server.

We also packaged the annotation tool and the website server to a single Docker image which is convenient for installing dependency libraries and reproducing experiments.

CHAPTER 7
CONCLUSION

In this chapter, we report the results in the process of the thesis's research. We also draw the future works for improving our proposed method and its applications.

7.1 Results

During the process of the thesis's research, we build a deep understanding of fundamental tasks in computer vision. We also learn the architecture of neural networks and convolutional neural networks. In this thesis's survey, we learn about state-of-the-art approaches for solving problems such as semantic segmentation and instance segmentation.

Specifically, we study the problem statement of video object segmentation and its challenges. We learn about how the interactive setting applies to the video object segmentation task. We also learn about datasets, interactive framework, evaluation metrics for interactive video object segmentation.

We propose a method for interactive video object segmentation that comprises three modules, including Control-point-based Scribbles-to-Mask, Self-Reference Refinement, and Guided Mask Propagation. We also experiment with different settings to choose the best configuration. Our proposed method satisfies all designed goals, including being fast, generating a good initial result, and improving accuracy after interactions. Our method achieves 0.767 and 0.759 in terms of Area Under the Curve (AUC) and $\mathcal{J}\&\mathcal{F}$ at 60 seconds ($\mathcal{J}\&\mathcal{F}$@60s), respectively, ranking 2^{nd} in the Interactive Scenario of The DAVIS Challenge 2020.

We also propose a video object segmentation annotation tool when annotators only have to annotate by scribbles. The time for generating masks is nearly real-time, and propagation speed is fast. We also propose a strategy that the

annotator only needs to annotate a subset of the frame instead of all the frames. Besides, we develop a semantic video website when videos are shown with semantic information of objects. In the process of implementing the annotation tool and the website, we learn about the framework that used to build a product Deep learning model, including Flask for API hosting, Gunicorn for the WSGI HTTP Server, Javascript for the web client. We also learn about the container technology that helps modules are portable and easy to reproduce and scale.

7.2 Future Works

There is a lot of space for our proposed method for interactive video object segmentation to increase accuracy. First, the scribbles return from the server from the second interaction onward can be better leveraged to refine previous predictions. Second, we need a better system to evaluate and decide which frames to be added to the list of candidates annotation frame in the next interaction. Finally, mask propagation should be guided by a better flow-based tracking method.

The annotation tool is running on the Qt platform. Although it is packaged in a Docker image, it still needs to run in a computer that runs the Linux operation system with CUDA available. In the future, we want to extend it and build it into the web platform. With that, people from all over the world can easily annotate on their Windows laptops or even their smartphones. We also want to build a lite version of our method that can run in the local client for the semantic website.

In the future, we want to build datasets with videos of arbitrary length. We will prepare high-resolution videos that have complicated cases such as object interaction, scale-invariant, and occlusions. For that purpose, we have to introduce a mechanism for fix minor errors in the predicted mask of interactive video object

segmentation by pixel-level annotation in case our interactive-based annotation tool cannot fix it.

And finally, we will build a large-scale media database about objects such as wild animals and people. This database will connect with our semantic video website to create a complete video education platform for kids that they can interact with objects in the video and learn information about these objects.

REFERENCES

[1] Stanford. Unsupervised feature learning and deep learning tutorial. 2017 (accessed July 13th, 2020). URL http://ufldl.stanford.edu/tutorial/supervised/MultiLayerNeuralNetworks/.

[2] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[3] Stanford. Cs231n: Convolutional neural networks for visual recognition. 2020 (accessed July 14th, 2020). URL https://cs231n.github.io/convolutional-networks/.

[4] DeepAI. Max pooling. 2019 (accessed July 14th, 2020). URL https://deepai.org/machine-learning-glossary-and-terms/max-pooling.

[5] Thalles Silva. Diving into deep convolutional semantic segmentation networks and deeplab_v3. 2018 (accessed Aug 4th, 2020). URL https://medium.com/free-code-camp/diving-into-deep-convolutional-semantic-segmentation-networks-and-deeplab-v3-4f094fa387df/.

[6] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[7] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[8] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3): 346–359, 2008.

[9] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4): 115–133, 1943.

[10] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[11] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[12] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[14] Paul Voigtlaender, Yuning Chai, Florian Schroff, Hartwig Adam, Bastian Leibe, and Liang-Chieh Chen. FEELVOS: fast end-to-end embedding learning for video object segmentation. *CoRR*, abs/1902.09513, 2019. URL `http://arxiv.org/abs/1902.09513`.

[15] Paul Voigtlaender, Jonathon Luiten, and Bastian Leibe. Boltvos: Box-level tracking for video object segmentation. *CoRR*, abs/1904.04552, 2019. URL http://arxiv.org/abs/1904.04552.

[16] Jonathon Luiten, Paul Voigtlaender, and Bastian Leibe. Premvos: Proposal-generation, refinement and merging for video object segmentation. In C. V. Jawahar, Hongdong Li, Greg Mori, and Konrad Schindler, editors, *Computer Vision - ACCV 2018 - 14th Asian Conference on Computer Vision, Perth, Australia, December 2-6, 2018, Revised Selected Papers, Part IV*, volume 11364 of *Lecture Notes in Computer Science*, pages 565–580. Springer, 2018. doi: 10.1007/978-3-030-20870-7_35. URL https://doi.org/10.1007/978-3-030-20870-7_35.

[17] S. Caelles, K.K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool. One-shot video object segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[18] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[19] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[20] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017.

[21] Minh-Triet Tran, Trung-Hieu Hoang, Tam V. Nguyen, Trung-Nghia Le, E-Ro Nguyen, Minh-Quan Le, Hoang-Phuc Nguyen-Dinh, Xuan-Nhat Hoang,

and Minh N. Do. Multi-referenced guided instance segmentation framework for semi-supervised video instance segmentation. *The 2020 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2020.

[22] I. E. Zulfikar, J. Luiten, and B. Leibe. Unovost: Unsupervised offline video object segmentation and tracking for the 2019 unsupervised davis challenge. *The 2019 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2019.

[23] S. Garg, V. Goel, and S. Kumar. Unsupervised video object segmentation using online mask selection and space-time memory networks. *The 2020 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2020.

[24] Y. Heo, Y. J. Koh, and C. Kim. Interactive video object segmentation using sparse-to-dense networks. *The 2019 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2019.

[25] Jiaxu Miao, Yunchao Wei, and Yi Yang. Memory aggregation networks for efficient interactive video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10366–10375, 2020.

[26] Konstantin Sofiiuk, Ilia Petrov, Olga Barinova, and Anton Konushin. f-brs: Rethinking backpropagating refinement for interactive segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8623–8632, 2020.

[27] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. URL `http://arxiv.org/abs/1312.6199`.

[28] Won-Dong Jang and Chang-Su Kim. Interactive image segmentation via backpropagating refinement scheme. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5297–5306, 2019.

[29] M. Tran, T. Le, T. V. Nguyen, T. Ton, T. Hoang, N. Bui, T. Do, Q. Luong, V. Nguyen, D. A. Duong, and M. N. Do. Guided instance segmentation framework for semi-supervised video instance segmentation. *The 2019 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2019.

[30] Sergi Caelles, Alberto Montes, Kevis-Kokitsi Maninis, Yuhua Chen, Luc Van Gool, Federico Perazzi, and Jordi Pont-Tuset. The 2018 davis challenge on video object segmentation. *arXiv:1803.00557*, 2018.

[31] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016.

[32] Yuk Heo, Yeong Jun Koh, and Chang-Su Kim. Interactive video object segmentation using global and local transfer modules. In *ECCV*, 2020. URL `https://openreview.net/forum?id=bo_lWt_aA`.

APPENDICES

LIST OF PUBLICATIONS

[1] Quoc-Cuong Tran, The-Anh Vu-Le, and Minh-Triet Tran. Interactive Video Object Segmentation with Multiple Reference Views, Self Refinement, and Guided Mask Propagation. *The 2020 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2020.

[2] Minh-Triet Tran, Thanh-An Nguyen, Quoc-Cuong Tran, Mai-Khiem Tran, Khanh Nguyen, Tu Ninh, Tu-Khiem Le, Hoang-Phuc Trang-Trung, Hoang-Anh Le, Hai-Dang Nguyen, Trong-Le Do, Viet-Khoa Vo-Ho, and Cathal Gurrin. FIRST - Flexible Interactive Retrieval SysTem for Visual Lifelog Exploration at LSC 2020. pages 67–72, 06 2020. doi: 10.1145/3379172.3391726.

[3] Minh-Triet Tran, Tam V. Nguyen, Trung-Hieu Hoang, Trung-Nghia Le, Khac-Tuan Nguyen, Dat-Thanh Dinh, Thanh-An Nguyen, Hai-Dang Nguyen, Xuan-Nhat Hoang, Trong-Tung Nguyen, Viet-Khoa Vo-Ho, Trong-Le Do, Lam Nguyen, Minh-Quan Le, Hoang-Phuc Nguyen-Dinh, Trong-Thang Pham, Xuan-Vy Nguyen, E-Ro Nguyen, Quoc-Cuong Tran, Hung Tran, Hieu Dao, Mai-Khiem Tran, Quang-Thuc Nguyen, Tien-Phat Nguyen, The-Anh Vu-Le, Gia-Han Diep, and Minh N. Do. iTASK - Intelligent Traffic Analysis Software Kit. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

Interactive Video Object Segmentation with Multiple Reference Views, Self Refinement, and Guided Mask Propagation

Quoc-Cuong Tran[1,3], The-Anh Vu-Le[1,3], and Minh-Triet Tran *[1,2,3]

[1]University of Science, VNU-HCM, Vietnam
[2]John von Neumann Institute, VNU-HCM, Vietnam
[3]Vietnam National University, Ho Chi Minh City, Vietnam

Abstract

In this work, we propose a fast and efficient method for interactive video object segmentation that takes scribbles as iterative input to specify the objects of interest and outputs the segmentation for those objects. Our approach contains three key modules: Control-point-based Scribbles-to-Mask, Self-Reference Refinement, and Guided Mask Propagation. It also maintains for each sequence a memory pool containing reliable frame-mask pairs, which is global across interactions, called Global Memory Pool. Additionally, we propose a strategy for the enrichment of this memory pool using more high-quality memory fragments, called Multiple Reference Views. Our proposed method achieved 0.767 and 0.759 in terms of Area Under the Curve (AUC) and $\mathcal{J}\&\mathcal{F}$ at 60 seconds ($\mathcal{J}\&\mathcal{F}$@60s), respectively, ranking 2nd in the Interactive Scenario of The DAVIS Challenge 2020.

1. Introduction

Video object segmentation aims to label pixel-level objects or background for a sequence of video frames. In an interactive scenario, a user gives iterative refinement inputs to the algorithm, in this case, in the form of scribbles, to segment the objects of interest.

In the first interaction of each sequence of frames, the server chooses a particular frame and provides a human-annotated scribble for each object in this frame. Based on these scribbles, the user's model has to predict segmentation masks of those objects for all the frames. After that, the user submits the predicted masks to the server. In each of the subsequent interactions, from a list of frames specified by the user, the server chooses the frame with the worst prediction and provides a new set of human-simulated scribbles in this frame. These scribbles point out the false positive and false negative regions. The user's model uses these human-simulated scribbles to refine its previous prediction masks. This procedure is repeated until a maximum number of interactions or a timeout is reached. The server measures the time needed to perform each interaction, these timings are combined to compute the final results [1].

Methods for tackling interactive video object segmentation challenge have to satisfy multiple goals, including being fast, generating a good initial result, and improving accuracy after interactions.

Our method view scribbles as a set of points. A Control-point-based Scribbles-to-Mask module is used to yield the mask of the scribbled frame from this set. The propagation phase usually is the bottle-neck on the runtime of video object segmentation methods. By using a memory-based model, our method can be fast, and the result improves by interactions. We use this memory-based model in the Self-Reference Refinement module for the mask achieved from the Control-point-based Scribbles-to-Mask module. To improve the memory pool of memory-based model to have a better reference, we propose a strategy for enriching the memory pool with reliable frames and Multiple Reference Views. We also implement a Guided Mask Propagation strategy to eliminate complex background for performance improvement.

The remainder of this paper is organized as follows. We show related work in Section 2. Our proposed methods are presented in Section 3. Experimental results are reported in Section 4. Finally, Section 5 concludes and paves the way for future work.

2. Related Work

In the Interactive Scenario of The DAVIS Challenge 2019, proposed methods from participants achieved promising results. The 1st place - Oh et al. [4], proposed

*Corresponding author. Email: tmtriet@fit.hcmus.edu.vn

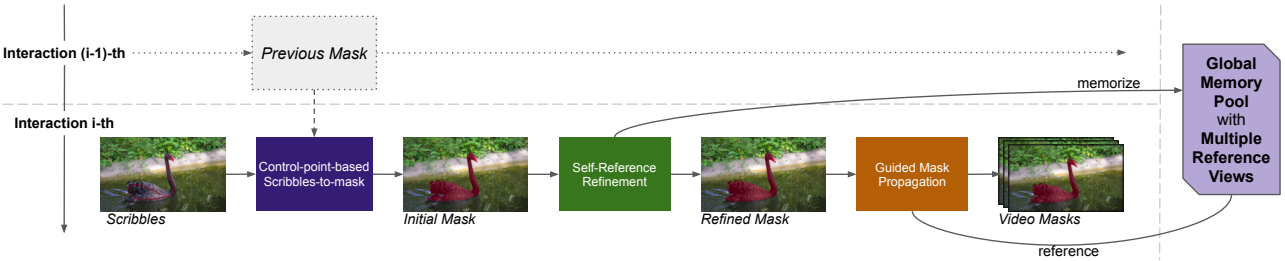

Figure 1. Overview of proposed method.

the Space-Time Memory Networks, which uses scribbled frames to form an external memory. This method is fast and shows the potential to improve the result after interactions. The 2nd place - Heo et al. [2], proposed a method of using two sparse-to-dense networks to yield the mask of scribbled frames and propagate points.

We implemented and extended the work of the following modules. For each object, f-BRS [3] takes into account two sets of points: points that belong to that object as positive points and points that do not belong to that object as negative points, to yield the mask. Space-Time Memory Networks [4] is fast, efficient, and can be extended with an enrichment strategy for the memory pool. To focus on the regions of interest of each object, a guided propagation strategy is promising to improve the result is suggested by Tran et al. [6].

3. Proposed Method

3.1. Pipeline

Figure 1 illustrates our proposed method. For each interaction, our method has two stages: interactive image object segmentation and video object mask propagation.

In the first stage, we develop a Control-point-based Scribbles-to-Mask module that converts input scribbles path into an initially predicted mask. After that, we apply on this initial mask the Self-Reference Refinement strategy, which is based on a memory-based model.

In the second stage, we propagate the self-refined mask using a memory-based model. This memory-based model utilizes a memory pool, called Global Memory Pool, which is filled with all frames having scribbles in current and previous interactions. Furthermore, we propose to enrich the Global Memory Pool with the Reliably Inferred Frames, obtained in the propagation process. In this way, we have more high-quality referenced views for the objects of interest. All frames are only influenced by its nearest reliable frame. We use a Guided Mask Propagation strategy to focus on potential regions of interest in a frame, as suggested in [6].

3.2. Control-point-based Scribbles-to-Mask

The input scribbles are first converted into control points using a Control-points Extractor. These control points are used by a Control-points-to-Mask module to generate a complete multi-object segmentation mask. If it is not the first interaction, a previously inferred mask is available and combined with this mask to generate the Initial Mask necessary for the next steps. Figure 2 shows an example of how it works.

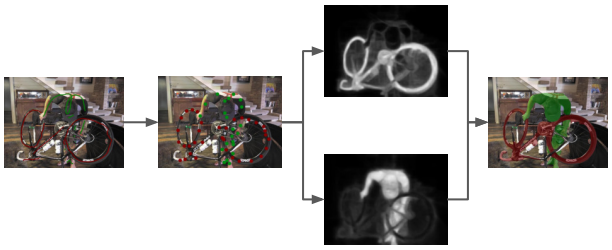

Figure 2. An example of Control-points-to-Mask.

Control-points Extractor To extract control points from a scribble path, we evenly sample points along that path. Each of these control points holds the information about its coordinates on the image and the object to which it belongs.

Control-points-to-Mask From the generated control points, we generate a probability mask for each object using a backpropagating refinement algorithm, called f-BRS [3]. This algorithm takes as input two sets of points, a positive set to localise the object position in the image, and a negative set to discriminate the object from other parts of the image. We consider control points belonging to that object positive points and control points belonging to other objects, often including the background class, as negative points. All the generated probability masks can be merged to generate the image object mask.

3.3. Global Memory Pool with Multiple Reference Views

The core algorithm used in the remaining components are based on a memory-based video object segmentation model. Memory-based models maintain a memory pool that

can be filled with frames and their corresponding masks. They perform segmentation on a frame with reference to that memory pool. We use such models in two different ways: as a refinement mechanism in the Self-Reference Refinement module and as a mask propagator in the Guided Mask Propagation module. In our work, we utilize the Space-Time Memory Networks [4] as the memory-based model.

For each sequence, our method maintains a memory pool that is global across interactions, called Global Memory Pool.

Naturally, our method relies heavily on the quality of the memory fragments stored in the Global Memory Pool. The most important fragments are the scribbled frames obtained after each interaction. Another set of frames to consider are the Reliably Inferred Frames: the predicted masks that are considered reliable obtained throughout the propagation process. These frames go through a filter to delete non-informative frames such as blurry, deformation frames, which are eventually enriched to improve memory pool quality.

3.4. Self-Reference Refinement

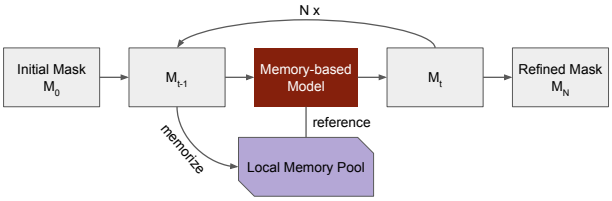

Figure 3. Self-Reference Refinement module.

As a refinement mechanism, the memory-based model maintains a Local Memory Pool that memorizes all previously inferred masks and uses that memory pool to perform segmentation on the same frame again (hence the name: Self-Reference). It is also possible to initialize the Local Memory Pool with the aforementioned Global Memory Pool. Figure 3 illustrates the Self-Refinement module.

Figure 4 shows examples where the Self-Refinement module helps refine the prediction.

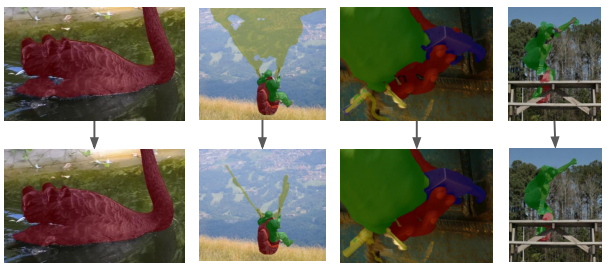

Figure 4. Cases that Self-Reference Refinement module helps improve the result: under-segmentation, over-segmentation.

3.5. Guided Mask Propagation

In each interaction, with the Refined Frame as the initiator, the propagation goes in two directions to both ends of the video sequence.

However, the process does not go all the way but instead at length calculated by considering the indices of the annotated frames in previous interactions. As can be seen from Figure 5, it stops midway between the current annotated frame index and the nearest previous annotated frame index. Considering the possible decay from the accumulated error of any propagation algorithm, this is a reasonable approach as each frame is only influenced by its nearest reliable frame.

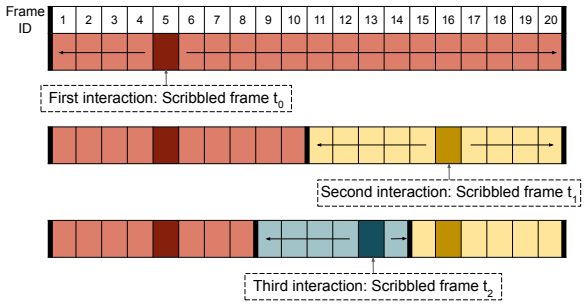

Figure 5. All frames are only influenced by its closest reliable frame.

To counter the problem of misidentification or over-segmentation, we employ a guidance mechanism in our propagation module. It uses the predicted masks of neighboring frames to generate a localisation mask. We use this mask to filter out potentially unrelated surroundings so that the segmentation algorithm can have a better focus on what to segment. Figure 6 shows an example where Guided Mask Propagation module focuses on regions of interest in a frame.

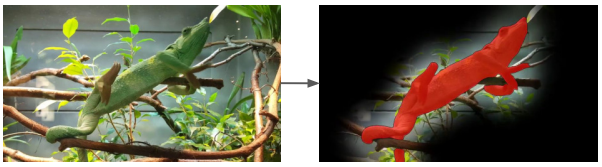

Figure 6. Region of Interest in a frame on sequence “chameleon”.

4. Experimental Results

For each sequence, more interactions mean more frames and their corresponding masks for the memory-based model to memorize, therefore we can expect the $\mathcal{J}\&\mathcal{F}$ score of our method to increase progressively after each interaction. Figure 8 shows our proposed method improves the $\mathcal{J}\&\mathcal{F}$ [5] score of sequences after interactions. About runtime, the method takes 25.72 seconds on average per interaction.

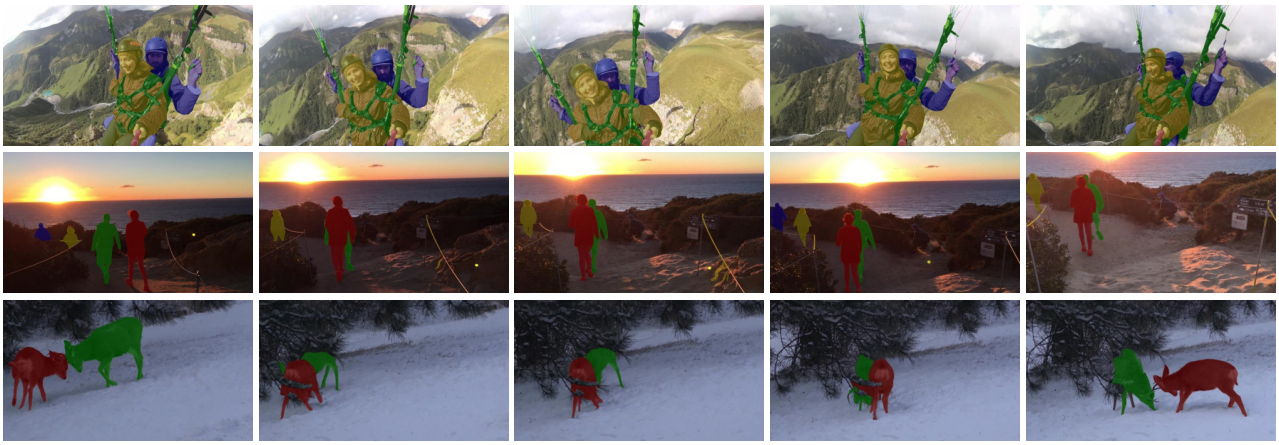

Figure 7. Final masks after the last interaction. Name of sequences from the first to the last row: “tandem”, “people-sunset”, and “deer”.

Figure 7 shows qualitative results of our proposed method. As shown in Table 1, our method achieves 0.767 and 0.759 in terms of Area Under the Curve (AUC) and $\mathcal{J}\&\mathcal{F}$ at 60 seconds ($\mathcal{J}\&\mathcal{F}$@60s), respectively, ranking 2^{nd} in Interactive Scenario of DAVIS Challenge 2020.

Table 1. Result on the DAVIS 2020 test-dev dataset.

Position	Participant	AUC	$\mathcal{J}\&\mathcal{F}$@60s
1	Yuk-Heo	0.772	0.787
2	**Ours**	**0.767**	**0.759**
3	ChenLIANG	0.754	0.762
4	Thanh-An Nguyen	0.468	0.473

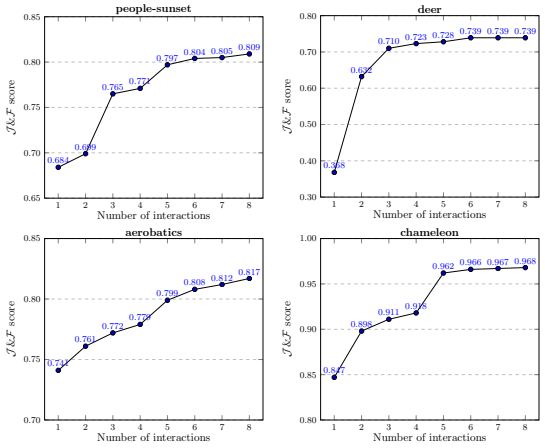

Figure 8. $\mathcal{J}\&\mathcal{F}$ score improving after interactions.

5. Conclusion

In this paper, we proposed a fast and efficient method for interactive video object segmentation that utilizes three key modules: Control-point-based Scribbles-to-Mask, Self-reference Refinement, and Guided Mask Propagation. The strong performance of our method is proven by the promising performance, both qualitatively and quantitatively, from the result of this year’s challenge.

We propose some directions for further research. First, the scribbles returned from the server from the second interaction onward can be better leveraged to refine previous predictions. Second, we need a better system to evaluate and decide which frames to be added to the memory pool. Third, mask propagation should be guided by a better, flow-based tracking method.

Acknowledgements

This research is supported by Vingroup Innovation Foundation (VINIF) in project code VINIF.2019.DA19. We also thank AIOZ Pte Ltd and the research group of Prof. Minh Do at University of Illinois at Urbana Champaign for supporting our research team with computing infrastructure.

References

[1] S. Caelles, A. Montes, K.-K. Maninis, Y. Chen, L. Van Gool, F. Perazzi, and J. Pont-Tuset. The 2018 davis challenge on video object segmentation. *arXiv:1803.00557*, 2018.

[2] Y. Heo, Y. J. Koh, and C. Kim. Interactive video object segmentation using sparse-to-dense networks. *The 2019 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2019.

[3] O. B. A. K. Konstantin Sofiiuk, Ilia Petrov. f-brs: Rethinking backpropagating refinement for interactive segmentation. *arXiv preprint arXiv:2001.10331*, 2020.

[4] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim. Video object segmentation using space-time memory networks. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[5] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016.

[6] M. Tran, T. Le, T. V. Nguyen, T. Ton, T. Hoang, N. Bui, T. Do, Q. Luong, V. Nguyen, D. A. Duong, and M. N. Do. Guided instance segmentation framework for semi-supervised video instance segmentation. *The 2019 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2019.

LSC '20, October 27, 2020, Dublin, Ireland
Proceedings published June 9, 2020

FIRST - Flexible Interactive Retrieval SysTem for Visual Lifelog Exploration at LSC 2020

Minh-Triet Tran[1,3,4], Thanh-An Nguyen[1,4], Quoc-Cuong Tran[1,4], Mai-Khiem Tran[1,4], Khanh Nguyen[1,4], Van-Tu Ninh[2], Tu-Khiem Le[2], Hoang-Phuc Trang-Trung [1,4], Hoang-Anh Le [1,4], Hai-Dang Nguyen[1,4], Trong-Le Do[1,4], Viet-Khoa Vo-Ho[1,4], Cathal Gurrin[2]
[1]University of Science, Ho Chi Minh City, Vietnam
[2]Dublin City University, Ireland
[3]John von Neumann Institute, Ho Chi Minh City, Vietnam
[4]Vietnam National University, Ho Chi Minh City, Vietnam

ABSTRACT

Lifelog can provide useful insights of our daily activities. It is essential to provide a flexible way for users to retrieve certain events or moments of interest, corresponding to a wide variation of query types. This motivates us to develop FIRST, a Flexible Interactive Retrieval SysTem, to help users to combine or integrate various query components in a flexible manner to handle different query scenarios, such as visual clustering data based on color histogram, visual similarity, GPS location, or scene attributes. We also employ personalized concept detection and image captioning to enhance image understanding from visual lifelog data, and develop an autoencoder-like approach for query text and image feature mapping. Furthermore, we refine the user interface of the retrieval system to better assist users in query expansion and verifying sequential events in a flexible temporal resolution to control the navigation speed through sequences of images.

CCS CONCEPTS

• **Information systems** → **Search interfaces**; *Multimedia databases*;
• **Human-centered computing** → Interactive systems and tools.

KEYWORDS

lifelog; interactive retrieval; information system; component integration

ACM Reference Format:
Minh-Triet Tran, Thanh-An Nguyen, Quoc-Cuong Tran, Mai-Khiem Tran, Khanh Nguyen, Van-Tu Ninh, Tu-Khiem Le, Hoang-Phuc Trang-Trung, Hoang-Anh Le,and Hai-Dang Nguyen, Trong-Le Do, Viet-Khoa Vo-Ho, Cathal Gurrin. 2020. FIRST - Flexible Interactive Retrieval SysTem for Visual Lifelog Exploration at LSC 2020. In *Proceedings of the Twenty-Fifth International Conference on Architectural Proceedings of the Third Annual Workshop on the Lifelog Search Challenge (LSC '20), June 9, 2020, Dublin, Ireland.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3379172.3391726

LSC '20, June 9, 2020, Dublin, Ireland

ACM ISBN 978-1-4503-7136-0/20/06…$15.00
https://doi.org/10.1145/3379172.3391726

1 INTRODUCTION

Lifelog [5] data provides valuable information for people to analyze their daily activities, events, and memories. However, it is not easy to describe what we want to search for from a massive amount of lifelog data, especially when much of the data is in a visual format. We can quickly enter a query in text format to retrieve text documents, but it is still a challenging problem to handle various types of queries, usually in text format, to look for a specific moment of interest in a collection of images or video clips.

The annual Lifelog Search Challenge (LSC [5]) aims to evaluate different approaches to assist users seeking to find events/moments of interest in an interactive manner from lifelog data. Many systems have been developed with different methods of image analysis and understanding, and different modalities for expressing queries and user interaction for result refinement [6].

Because of the wide variation of query types, such as location or scene-based queries, temporal sequence queries, or concept-based queries, the retrieval system should be flexible enough to add more query processing strategies or pipelines, or even support users to define their customized searching workflows. This motivates our proposal for a Flexible Interactive Retrieval SysTem (FIRST) to support different types of queries and to be open for future extensions.

Our proposed system is an enhancement to an existing system previously developed for the LSC2019 [10] with the following new features. Firstly, we refactor our legacy system and develop an integration platform that can be used to define and execute new query workflows and visualization layouts. For example, we can now visualize images into clusters with different semantic criteria, such as color histograms, scene attributes, or extracted deep features. Secondly, we improve our methods for image and scene understanding. We leverage our personalization strategy to generate captions for images by adapting our image captioning module to the personal lifelog data. We also propose an autoencoder-like method for mapping the features of a text query and an image to a common space to measure its semantic relationship. Thirdly, besides the default legacy query layout[10] , we create different visual layouts with clusters of images to assist users in searching for pictures in groups and add more interactions for users, such as query expansion by positive and negative examples. We also refine the user interface in [10] to better assist users in searching and verifying sequential events with flexible temporal resolution.

The content of this paper is organized as follows. In Section 2, we briefly review related approaches for lifelogging retrieval. We introduce an overview of our system in Section 3 with the key idea to create a flexible integration platform to define and execute different pipelines. We introduce main methods to extract information from images in Section 4, including personalized concept detection and caption generation, as well as scene text extraction. The query layouts and interactions are presented in Section 5. The conclusion and discussion for future work are in Section 6.

2 RELATED WORK

Lifelog analysis and retrieval has become an attractive research topic in recent years. The first goal is to propose better methods for image/video understanding, and a second one is to develop more convenient modalities for users to interact with query systems. Challenges in different formats have been organized, targeting these two goals. In the recent ImageCLEF lifelog task [1] and the recent NTCIR14-Lifelog task [4], the retrieval tools are used by their creators who have in-depth knowledge about their systems, and the objective for participants mainly focuses on how to better extract information from visual lifelog data. In Lifelog Search Challenge (LSC [5]), the goals are not only to improve image understanding but also to design and enhance usability for professional and novice users to interact with the retrieval systems to handle various query types.

Successful systems have been developed and used in the Lifelog Search Challenge and the Visual Browser Showdown (VBS) [20]. The vitrivr system [19] supports different media types, such as images, videos, and audios, and also groups image sequences into segments for better representation. VIRET system [16] utilizes both visual data and non-visual information, such as weekdays, biometric data, GPS locations, to assist users in filtering certain events of interest. The SOM Hunter system [9] employs the technique proposed by Xirong Li[14] to compare the semantic distance between a text query and a video clip, which can be adapted for text query and image distance evaluation. Image clustering are also used in VIREO [18] and SOM-Hunter [9] systems. The system of the HCMUS team at LSC 2019 [12] enhanced the metadata by defining personalized visual concepts and creates an interface to navigate images in a sequence for temporal event verification. The Lifeseeker system [13] employed the Bag-of-Words model for text retrieval and query expansion using a Word2Vec model [17] pre-trained on GoogleNews dataset.

3 OVERVIEW OF FLEXIBLE INTERACTIVE RETRIEVAL SYSTEM

3.1 System Overview

Figure 1 illustrates the overview of our retrieval system. The essential feature in the system is the Flexible Integration Platform (see Section 3.2), which can be used to (1) define different component interaction as pipelines; (2) execute the workflow defined in a pipeline; and (3) convert and bridge data between components. This platform serves the other three main subsystems: Layout Manager, Query Component Manager, and MetaData Manager.

The MetaData Manager subsystem uses both visual and non-visual information, such as GPS data. For visual information, besides

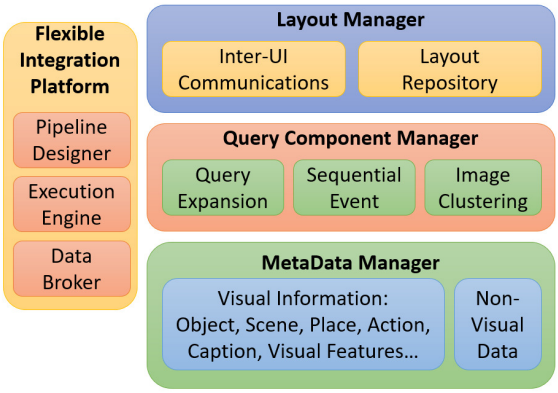

Figure 1: System overview of the query system

.

the concepts provided by the organizers, such as concepts extracted from Mask-RCNN [7] or scene attributes extracted using Place API, we extract concepts using CenterNet [3] and create a pipeline to train customized object detectors (see Section 4.1). We also refine the image captioning method [23] to adapt to the lifelog data.

The Query Component Manager subsystem is responsible for processing different query types. Users can use the query expansion module to find and select concepts related to the query based on word embedding distance. The sequential event module exploits the temporal event relationship and frequent sequential events. The image clustering module helps users in interacting with visual data with different semantic distances, such as color histograms, deep visual features, or GPS locations, etc. Besides, in this subsystem, we can add more future-defined query processing pipelines using the Flexible Integration Platform.

In the Layout Manager subsystem, we create a repository of different layouts, including the traditional query interface [10], image cluster visualization layouts, etc. We also create a communication mechanism between different UI components to allow flexible inter-UI interaction.

3.2 Flexible Integration Platform

By using an integration platform, we can customize the process to handle various query types. Users can select the workflows or pipelines that they think the most suitable for the problem by choosing available components and define how these components connect to others. Users can communicate to the integration platform efficiently by dragging and dropping interaction, then execute the defined pipeline to process queries.

The platform consists of a diagram library for an end-user to quickly create a runnable workflow with minimal effort and to monitor the process when running. On the back-end side, we have an engine for managing the workflow, including component integration, data flow control, logging.

Each component has to support to notify the current state: initializing, ready, executing, finished. When a component finishes, its next components in the pipeline receive the output as their inputs to begin their execution.

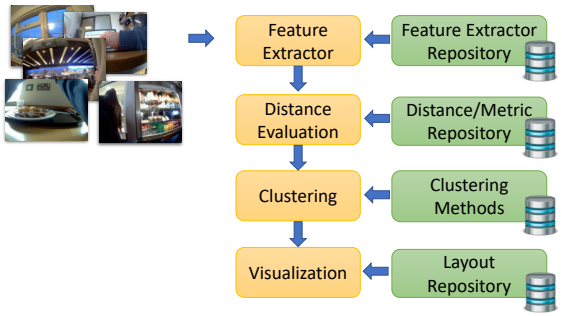

Figure 2: Pipeline for visual/semantic distance estimation and visualization.

We define multiple atomic components for different tasks: feature extractors, image clustering, visualizer, etc. Components that support the same mission have the same format for input and output. Workflow data come from various multiple forms, such as images, videos, vector, etc. We also pre-define several common best practice flows from our experience that can be selected for use by users.

Our platform also supports the workflow with parallel branches by using if-condition. For example, we can check if the location vector has 2-dimension, then we can use a grid visualizer, but if it has 3-dimension or more, we can use VR for visualization.

3.3 Flexible Image Clustering

In [22], we proposed to group images based on BoW- similarity. In this way, we can have clusters of images, and each may correspond to a working place. From this strategy, we continue to develop a more flexible image clustering technique. Our extension is also inspired by image clustering functions in VIREO [18] and SOM-Hunter [9] systems.

Figure 2 illustrates our pipeline to create different strategies to estimate the visual/semantic distance between images. By using our integration system, we can easily change and test with other feature extractors such as EfficientNet [21] or ResNet [8]. We also can test will multiple dimension reduction algorithms, such as PCA. For example, we use ResNet50 with weights trained on ImageNet to extract features and multi-dimensional scaling with pairwise Euclidean distance for dissimilarity measure to reduce the dimension to 2D.

4 INFORMATION EXTRACTION FROM IMAGES

4.1 Personalized Concept Detection

In our previous work [10, 11], we proposed to detect and extract personalized visual concepts from visual lifelog data. We first manually extract personalized concepts of a lifelogger, i.e., his or her everyday visual objects that are not available in public visual datasets, and train object detectors using Faster-RCNN. In this way, our system can better adapt to the personal life of a lifelogger and can detect such visual concepts from his or her lifelog data.

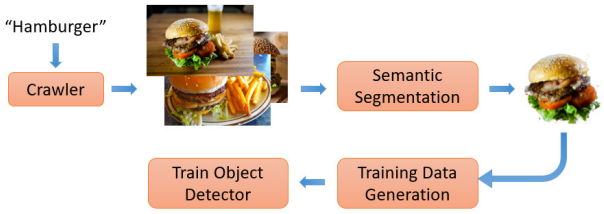

Figure 3: Pipeline for creating personalized object detector.

In this paper, we take further steps to enhance the personalization for lifelog analysis. Firstly, we create a pipeline to automatically crawl images from the Internet which are corresponding to a new visual concept, such as c̈offee machineör ḧamburger; extract the main content using semantic segmentation, embed objects into different backgrounds, train object detectors, and apply the detectors on lifelog dataset. In this way, we can boost up the process to define numerous personalized object detectors adapting to the lifelogger's habits (see Figure 3).

4.2 Personalized Caption Generation

Figure 4: An example of a personalized caption.

We utilize our proposed method for concept-augmented image captioning [23] to train a captioning module based on a small dataset of captioning for a lifelogger. We randomly select a subset of 1,000 images from the lifelog dataset and manually annotate their captions. We have 2-3 volunteers annotating each image. In this way, we create a small but sufficient volume of data to refine our captioning module. An example of generated personalized captions is shown in Figure 4. Then we use our refined captioning module to generate captions for the whole lifelog dataset. The dense captioning strategy is also used to create more descriptions for different regions in an image. The generated captions are stored in our database to be matched against a future query text or phrase.

4.3 Scene Text Extraction

To better understand information from an image, we use Convolutional Character Networks [24] to extract scene texts from an image, such as brand names of products in a shop, shop names, street names, etc. We also use ABCNet [15] to handle texts in Bezier-Curve shapes. Figure 5 shows two examples of extracted texts from images.

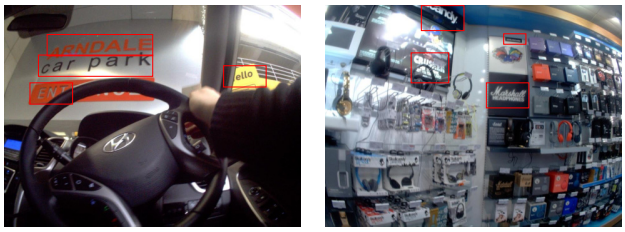

Figure 5: An example of a personalized caption.

As recognized text can be occluded, we use a dictionary to auto-correct and the edit distance to compare the similarity between a query text and extracted text from an image. Furthermore, we employ Word2Vec to evaluate the semantic relationship between words/phrases.

4.4 Mapping Query Text and Images to An Invertible Common Feature Space with AutoEncoder

To compare the similarity between a query text and an image, we can encode the query and the image into the sentence and image features, respectively. We can use BERT for sentence embedding, and ResNet for image embedding. A common approach is to map these two features into a common space to measure the distance between them, then infer the relationship between the query text and the image [14].

Figure 6 shows the overview of our proposed method to map a query text of an image to a common feature space with autoencoder. An important property of the common space is that it can be used to measure the distance or dissimilarity between the features of two different data types (or domains).

In our proposed method, we aim to achieve one more property for the common space: the information of a feature in the common space should be enough to approximately reconstruct the origin feature, either from a text query or an image. Therefore, together with two mapping functions from the sentence embedding and the image embedding to the common space, we also define the two

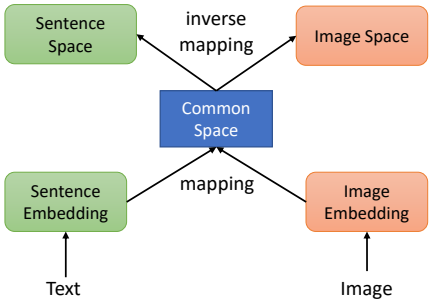

Figure 6: AutoEncoder-like approach for mapping query text and image to a common feature space

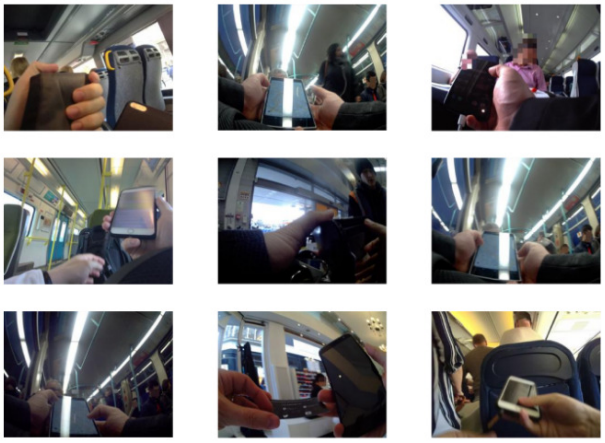

Figure 7: Retrieved images for "using cellphone in a train".

inverse mapping functions from the common space back to the feature spaces for sentences and images.

Our auto-encoder like approach is inspired from the idea of the dual encoding for zero-example video retrieval [2]. In Figure 7, we demonstrate the result of the query to search for "using cellphone in a train" using our proposed method for text and image mappings with autoencoder.

5 QUERY LAYOUT AND INTERACTION

5.1 Query Layouts

The presentation subsystem plays an essential part in visualizing the results intuitively because a clear representation of data can have a significant impact on the insights derived from queries.

With expandability and usability in mind, we build the presentation subsystem as a collection of modules working in tandem, processing, and rendering queries from the source to the results. As such, it covers several configurable parameters.

While the sheer number of parameters provides flexibility over the system, it also has the potential to worsen the usability due to its complexity. To avoid such problems, we have several presets which are preloaded that suit basic visualizing scenarios.

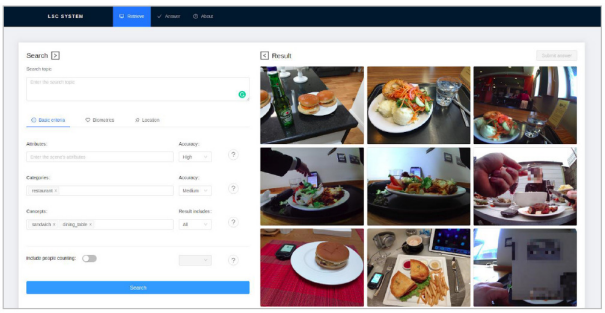

Figure 8: Legacy layout as the default mode [11]

Our current system is developed from our previous retrieval system [10]. For the LSC in 2020, we reuse our query interface in [11], as shown in Figure 8. We keep the default layout because of its simplicity for users to use, but we add more interaction functions into the system.

In the default mode, users can input keywords or phrases to query for a moment. Together with new techniques to extract more concepts from images (see Section 4), we also adopt the semantic similarity between a query phrase and an image. Our method is inspired by the work of Xirong Li et al. [14]. The key idea is to encode a query phrase as a feature in the feature space of images. In this way, we can measure the distance between the encoded feature of a query phrase and the visual feature of an image. We also propose a new method for this function in Section 4.4.

5.2 Image Cluster Visualization

In Figure 9, we illustrate the layout in our system to visualize images based on their semantic similarity. In our system, the semantic similarity can be determined in different ways, such as GPS, Bag-of-Visual-Word features, scene attributes, or color histograms.

We also support grouping a sequence of consecutive images into an event, an allow users to change the level of details in the image cluster visualization layout. In the global view, images are grouped in several main clusters. When a user selects an image cluster, it is expanded to show its subgroups. A user can explore image clusters in a coarse-to-fine approach.

Figure 9: Image Visualization based on Semantic Similarity.

5.3 Example-based Query Expansion

It would be a convenient way to assist users in finding moments in a lifelog data having similar content with a sample image. For example, when we find an example image of watching TV at home, we can use this image as a positive example of finding all similar moments.

In this way, we can replace multiple query criteria to find the example image by that image to further retrieve other images or moments. We also allow users to select an image as a negative example to eliminate images that are similar to it.

For any image displayed in the user interface of our system, we provide a query expansion mechanism from that image. A selected image is considered as a positive or negative example for query expansion. As illustrated in Figure 10, we want to find the moments of watching TV, and the TV must be turned on. In the retrieved list, images or moments are retrieved and sorted in descending order of similarity with positive examples (in the green box: TV is on), then in ascending order of dissimilarity with negative examples (in the red box: TV is off).

Figure 10: Example-based query expansion to find images/moments similar or different from examples.

5.4 Flexible Temporal Resolution for Sequence of Events Exploration

In [10], we created a user interface to assist users in surfing a sequence of images (backward and forward) from any given moment in the lifelog data to look for certain other events before or after the selected moment. However, in this implementation, we notice that it is time-consuming for a user to navigate to a long-distance event as we use a fixed time step. Thus, in the current system, we allow users to easily adjust their navigation step by using a temporal step slider. We expect this way can help users in controlling the operation that best matches their needs.

6 CONCLUSION AND FUTURE WORK

In this paper, we introduced our flexible system for lifelog data retrieval. The key idea of our system is to enhance the flexibility to define different pipelines to handle both visual data understanding and query processing.

We use the designer in Flexible Integration Platform to define then execute various workflows that might be useful for a wide variation of query types. We also leverage the personalization in visual lifelog data analysis with adapted image captioning, and utilize the text and visual distance comparison [14] to retrieve images with captions semantically related to a text query.

Currently, based on our subjective experience, we predefined several pipelines to illustrate the usability of the Flexible Integration Pipeline in image analysis and query processing. We expect that the flexible mechanism of our system can efficiently assist researchers and users in developing and integrating their own components into the platform and defining then executing their customized workflows for multimedia analysis and retrieval.

ACKNOWLEDGMENTS

This research is partially supported by Vingroup Innovation Foundation (VINIF) in project code VINIF.2019.DA19 and Vietnam Ireland Bilateral Education Exchange Programme (VIBE) 2019. We would like to thank AIOZ Pte Ltd for supporting our research team with computing infrastructure.

REFERENCES

[1] Duc-Tien Dang-Nguyen, Luca Piras, Michael Riegler, Minh-Triet Tran, Liting Zhou, Mathias Lux, Tu-Khiem Le, Van-Tu Ninh, and Cathal Gurrin. 2019. Overview of ImageCLEFlifelog 2019: Solve my life puzzle and Lifelog Moment Retrieval. In *CLEF2019 Working Notes (CEUR Workshop Proceedings)*. CEUR-WS.org <http://ceur-ws.org>, Lugano, Switzerland.

[2] Jianfeng Dong, Xirong Li, Chaoxi Xu, Shouling Ji, Yuan He, Gang Yang, and Xun Wang. 2018. Dual Encoding for Zero-Example Video Retrieval. arXiv:cs.CV/1809.06181

[3] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. 2019. CenterNet: Keypoint Triplets for Object Detection. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*. IEEE, 6568–6577. https://doi.org/10.1109/ICCV.2019.00667

[4] Cathal Gurrin, H. Joho, Frank Hopfgartner, Liting Zhou, Tu Ninh, Tu-Khiem Le, Rami Albatal, D.-T Dang-Nguyen, and Graham Healy. 2019. Overview of the NTCIR-14 Lifelog-3 task.

[5] Cathal Gurrin, Tu-Khiem Le, Van-Tu Ninh, Duc-Tien Dang-Nguyen, Björn Þór Jónsson, Jakub Lokoč, Wolfgang Hurst, Minh-Triet Tran, and Klaus Schoeffmann. 2020. An Introduction to the Third Annual Lifelog Search Challenge, LSC'20. In *ICMR '20, The 2020 International Conference on Multimedia Retrieval*. ACM, Dublin, Ireland.

[6] Cathal Gurrin, Klaus Schoeffmann, Hideo Joho, Liting Zhou, Aaron Duane, Andreas Leibetseder, Michael Riegler, Luca Piras, Minh-Triet Tran, Jakub Lokoč, and Wolfgang Hürst. 2019. Comparing Approaches to Interactive Lifelog Search at the Lifelog Search Challenge (LSC2018). *ITE Transactions on Media Technology and Applications* 7, 2 (2019), 46–59.

[7] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. 2017. Mask R-CNN. *CoRR* abs/1703.06870 (2017). arXiv:1703.06870 http://arxiv.org/abs/1703.06870

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 770–778. https://doi.org/10.1109/CVPR.2016.90

[9] Miroslav Kratochvíl, Patrik Veselý, Frantisek Mejzlík, and Jakub Lokoc. 2020. SOM-Hunter: Video Browsing with Relevance-to-SOM Feedback Loop. In *MultiMedia Modeling - 26th International Conference, MMM 2020, Daejeon, South Korea, January 5-8, 2020, Proceedings, Part II (Lecture Notes in Computer Science)*, Yong Man Ro, Wen-Huang Cheng, Junmo Kim, Wei-Ta Chu, Peng Cui, Jung-Woo Choi, Min-Chun Hu, and Wesley De Neve (Eds.), Vol. 11962. Springer, 790–795. https://doi.org/10.1007/978-3-030-37734-2_71

[10] Nguyen-Khang Le, Dieu-Hien Nguyen, Trung-Hieu Hoang, Thanh-An Nguyen, Thanh-Dat Truong, Tung Dinh Duy, Quoc-An Luong, Viet-Khoa Vo-Ho, Vinh-Tiep Nguyen, and Minh-Triet Tran. 2019. Smart Lifelog Retrieval System with Habit-based Concepts and Moment Visualization. In *Proceedings of the ACM Workshop on Lifelog Search Challenge, LSC@ICMR 2019, Ottawa, ON, Canada, 10 June 2019*, Cathal Gurrin, Klaus Schöffmann, Hideo Joho, Duc-Tien Dang-Nguyen, Michael Riegler, and Luca Piras (Eds.). ACM, 1–6.

[11] Nguyen-Khang Le, Dieu-Hien Nguyen, Vinh-Tiep Nguyen, and Minh-Triet Tran. 2019. Lifelog Moment Retrieval with Advanced Semantic Extraction and Flexible Moment Visualization for Exploration. In *Working Notes of CLEF 2019 - Conference and Labs of the Evaluation Forum, Lugano, Switzerland, September 9-12, 2019 (CEUR Workshop Proceedings)*, Linda Cappellato, Nicola Ferro, David E. Losada, and Henning Müller (Eds.), Vol. 2380. CEUR-WS.org. http://ceur-ws.org/Vol-2380/paper_139.pdf

[12] Nguyen-Khang Le, Dieu-Hien Nguyen, Trung-Hieu Hoang, Thanh-An Nguyen, Thanh-Dat Truong, Duy-Tung Dinh, Quoc-An Luong, Viet-Khoa Vo-Ho, Vinh-Tiep Nguyen, and Minh-Triet Tran. 2019. Smart Lifelog Retrieval System with Habit-Based Concepts and Moment Visualization. Association for Computing Machinery, New York, NY, USA.

[13] Tu-Khiem Le, Van-Tu Ninh, Duc-Tien Dang-Nguyen, Minh-Triet Tran, Liting Zhou, Pablo Redondo, Sinéad Smyth, and Cathal Gurrin. 2019. LifeSeeker: Interactive Lifelog Search Engine at LSC 2019. In *Proceedings of the ACM Workshop on Lifelog Search Challenge, LSC@ICMR 2019, Ottawa, ON, Canada, 10 June 2019*, Cathal Gurrin, Klaus Schöffmann, Hideo Joho, Duc-Tien Dang-Nguyen, Michael Riegler, and Luca Piras (Eds.). ACM, 37–40. https://doi.org/10.1145/3326460.3329162

[14] Xirong Li, Chaoxi Xu, Gang Yang, Zhineng Chen, and Jianfeng Dong. 2019. W2VV++: Fully Deep Learning for Ad-hoc Video Search. In *Proceedings of the 27th ACM International Conference on Multimedia, MM 2019, Nice, France, October 21-25, 2019*, Laurent Amsaleg, Benoit Huet, Martha A. Larson, Guillaume Gravier, Hayley Hung, Chong-Wah Ngo, and Wei Tsang Ooi (Eds.). ACM, 1786–1794.

[15] Yuliang* Liu, Hao* Chen, Chunhua Shen, Tong He, Lianwen Jin, and Liangwei Wang. 2020. ABCNet: Real-time Scene Text Spotting with Adaptive Bezier-Curve Network. In *Accepted to Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2020*.

[16] Jakub Lokoč, Tomáš Souček, Premysl Čech, and Gregor Kovalčík. 2019. Enhanced VIRET Tool for Lifelog Data. Association for Computing Machinery, New York, NY, USA.

[17] Tomas Mikolov, G.s Corrado, Kai Chen, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. 1–12.

[18] Phuong Anh Nguyen, Jiaxin Wu, Chong-Wah Ngo, Danny Francis, and Benoit Huet. 2020. VIREO @ Video Browser Showdown 2020. In *MultiMedia Modeling - 26th International Conference, MMM 2020, Daejeon, South Korea, January 5-8, 2020, Proceedings, Part II (Lecture Notes in Computer Science)*, Yong Man Ro, Wen-Huang Cheng, Junmo Kim, Wei-Ta Chu, Peng Cui, Jung-Woo Choi, Min-Chun Hu, and Wesley De Neve (Eds.), Vol. 11962. Springer, 772–777. https://doi.org/10.1007/978-3-030-37734-2_68

[19] Luca Rossetto, Ralph Gasser, Silvan Heller, Mahnaz Amiri Parian, and Heiko Schuldt. 2019. Retrieval of Structured and Unstructured Data with Vitrivr. Association for Computing Machinery, New York, NY, USA.

[20] K. Schoeffmann. 2019. Video Browser Showdown 2012-2019: A Review. In *2019 International Conference on Content-Based Multimedia Indexing (CBMI)*. 1–4. https://doi.org/10.1109/CBMI.2019.8877397

[21] Mingxing Tan and Quoc V. Le. 2019. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.), Vol. 97. PMLR, 6105–6114. http://proceedings.mlr.press/v97/tan19a.html

[22] Thanh-Dat Truong, Tung Dinh Duy, Vinh-Tiep Nguyen, and Minh-Triet Tran. 2018. Lifelogging Retrieval based on Semantic Concepts Fusion. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge, LSC@ICMR 2018, Yokohama, Japan, June 11, 2018*, Cathal Gurrin, Klaus Schoeffmann, Hideo Joho, Duc-Tien Dang-Nguyen, Michael Riegler, and Luca Piras (Eds.). ACM, 24–29. https://doi.org/10.1145/3210539.3210545

[23] Viet-Khoa Vo-Ho, Quoc-An Luong, Duy-Tam Nguyen, Mai-Khiem Tran, and Minh-Triet Tran. 2018. Personal Diary Generation from Wearable Cameras with Concept Augmented Image Captioning and Wide Trail Strategy. In *Proceedings of the Ninth International Symposium on Information and Communication Technology, SoICT 2018, Danang City, Vietnam, December 06-07, 2018*. ACM, 367–374.

[24] Linjie Xing, Zhi Tian, Weilin Huang, and Matthew R Scott. 2019. Convolutional Character Networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

iTASK - Intelligent Traffic Analysis Software Kit

Minh-Triet Tran*[1,2,3], Tam V. Nguyen[4], Trung-Hieu Hoang[1,2], Trung-Nghia Le[5], Khac-Tuan Nguyen[1,2], Dat-Thanh Dinh[1,2], Thanh-An Nguyen[1,2], Hai-Dang Nguyen[1,2], Xuan-Nhat Hoang[1,2], Trong-Tung Nguyen[1,2], Viet-Khoa Vo-Ho[1,2], Trong-Le Do[1,2], Lam Nguyen[1,2], Minh-Quan Le[1,2], Hoang-Phuc Nguyen-Dinh[1,2], Trong-Thang Pham[1,2], Xuan-Vy Nguyen[1,2], E-Ro Nguyen[1,2], Quoc-Cuong Tran[1,2], Hung Tran[1,2], Hieu Dao[1,2], Mai-Khiem Tran[1,2], Quang-Thuc Nguyen[1,2], Tien-Phat Nguyen[1,2], The-Anh Vu-Le[1,2], Gia-Han Diep[1,2], and Minh N. Do[6]

[1]University of Science, Ho Chi Minh City, Vietnam
[2]Vietnam National University, Ho Chi Minh City, Vietnam
[3]John von Neumann Institute, Ho Chi Minh City, Vietnam
[4]University of Dayton, U.S.
[5]National Institute of Informatics, Japan
[6]University of Illinois at Urbana-Champaign, U.S.

Abstract

Traffic flow analysis is essential for intelligent transportation systems. In this paper, we introduce our Intelligent Traffic Analysis Software Kit (iTASK) to tackle three challenging problems: vehicle flow counting, vehicle re-identification, and abnormal event detection. For the first problem, we propose to real-time track vehicles moving along the desired direction in corresponding motion-of-interests (MOIs). For the second problem, we consider each vehicle as a document with multiple semantic words (i.e., vehicle attributes) and transform the given problem to classical document retrieval. For the last problem, we propose to forward and backward refine anomaly detection using GAN-based future prediction and backward tracking completely stalled vehicle or sudden-change direction, respectively. Experiments on the datasets of traffic flow analysis from AI City Challenge 2020 show our competitive results, namely, S1 score of 0.8297 for vehicle flow counting in Track 1, mAP score of 0.3882 for vehicle re-identification in Track 2, and S4 score of 0.9059 for anomaly detection in Track 4. All data and source code are publicly available on our project page. [1]

1. Introduction

Traffic analysis is an essential component in any AI city worldwide. There are several problems related to traffic analysis such as vehicle type classification [18, 35], vehicle localization [13, 44] , velocity estimation [10, 14], vehicle tracking [5], car fluent recognition [20], vehicle re-identification [1, 22, 32], or abnormal event detection [19, 30, 46]. In this paper, we focus on three challenging problems in the real world presented in AI City Challenge 2020, namely vehicle flow counting, vehicle re-identification, and anomaly detection.

We propose an Intelligent Traffic Analysis Software Kit (iTASK) to tackle these three problems:

- Real-time vehicle flow counting: we propose to represent each motion-of-interest (MOI) as a corresponding non-overlapped region-of-interest (ROI) to track vehicles moving along the desired direction. These non-overlapped ROIs are selected so as to reduce the possibility to (1) lose tracking a vehicle and (2) be confused between nearby MOIs.
- Vehicle re-identification: we propose to restate the re-identification problem into the document retrieval with bags of vehicle attributes. We define multiple vehicle attribute analyzers for scene text, logos, wheel types, view types, front and rear light types. An image of a vehicle is now represented as a document with multiple semantic words; each corresponds to a vehicle attribute.

*Corresponding author. Email: tmtriet@fit.hcmus.edu.vn
[1]https://github.com/selab-hcmus/AI_City_2020

- Anomaly detection: We propose forward and backward refinements for anomaly event detection. For forward prediction, we use UNet GAN to generate a future frame from the current frame and its accumulative motion-blend data, then check the generated frame against the real next frame to see if there is a significant difference between them. For backward tracking, we track a detected stalled vehicle to refine the moment when it begins to stop completely or begins to move in a sudden-change direction.

We achieve promising results on AI City Challenge 2020. In track 1 for vehicle flow counting, we achieve the S1 score of 0.8297, the 10^{th} place out of 18 team submission. In Track 2 for vehicle re-identification, we achieve 0.3882 on mAP, the 26^{th} place out of 41 team submissions. In Track 3 for anomaly detection, we take the 5^{th} place out of 13 team submissions with F1 score of 0.9421 and RMSE of 11.2556.

The remainder of this paper is organized as follows. Section 2, we briefly review related work. We then present our solutions for real-time vehicle flow detection and counting, vehicle re-identification with attribute sets, and anomaly event detection and refinement in Section 3, 4, and 5, respectively. Experimental resutls on Track 1, 2, and 4 of AI City Challenge 2020 are then reported and discussed in Section 6. Finally, Section 7 draws the conclusion.

2. Related Work

AI City Challenge [25, 26, 39] in recent years has promoted many challenging problems of traffic video analysis such as vehicle counting, velocity estimation, behavior analysis, vehicle re-identification, and anomaly detection. Here, we briefly review the tasks of vehicle flow counting, vehicle re-identification, and anomaly detection.

Different from traditional vehicle counting, which is old fashion and identifies vehicles based on only their appearance, vehicle flow counting is a new problem. Vehicles are identified based on their movements, including flow and direction of motion. Therefore, techniques of object detection and object tracking are combined to solve the problem. Several techniques, such as multiple adaptive detectors [40, 28], scanline based on landmarks [40], graph matching [45], and geometric calibration based estimation [33] have been utilized and achieved the high performance.

Vehicle re-identification is a challenging problem that attracts research communities recently. Different from person re-identification, in which a person can be identified by his/her appearance, vehicle re-identification is more challenging as vehicles' appearance are usually nearly similar, especially for same-brand vehicles. To re-identify vehicles, triplet loss based deep learning metrics [38, 49, 21, 11, 15, 36, 4] are popularly applied to learn vehicle feature representation. Besides, spatial verification [28] and metadata distance [12] are used as post-processing to re-rank results from deep metric embedding networks.

Traffic anomaly detection is drawing attention from the research community [30, 34]. Several methods have been proposed to detect anomaly in traffic cameras effectively, such as GAN-based future prediction to capture contextual motion [27], background modeling techniques to eliminate moving vehicles based on average image [46, 28] and deep network [23], anomaly detection based on velocity estimation [46] and vehicle trajectories [48], attention-based model [17], and motion mask region [42].

3. Vehicle Flow Counting with Refined Motion-specific Region-of-Interest

3.1. Overview

The objective of vehicle flow counting is to determine the number of vehicles in each vehicle type moving in some specific motion-of-interest (MOI) in a global region-of-interest (ROI) in a traffic video from a fixed camera.

Figure 1 illustrates the overview of our proposed solution for realtime vehicle flow counting. Our pipeline comprises three main phases. First, we define a collection of adaptive vehicle detectors to handle different detection contexts, such as day or night environments, regular or tiny instances, etc. Second, we use expanded IoU to track vehicle instances across frames with a step-size k frames to reduce computational cost while maintaining acceptable accuracy. Last, based on the given motion-of-interest (MOI), we define multiple reliable motion-specific region-of-interest (m-ROIs) to efficiently track and count vehicles in each flow and to reduce the possibility of confusion between vehicles in similar motion flows.

We employ the adaptive vehicle detector scheme [40] to train detectors for two main classes: car and truck. We randomly select 1000 frames from different video clips in Track 1 of AI City Challenge 2020 for vehicle annotation and train various detectors for two main classes: vehicle

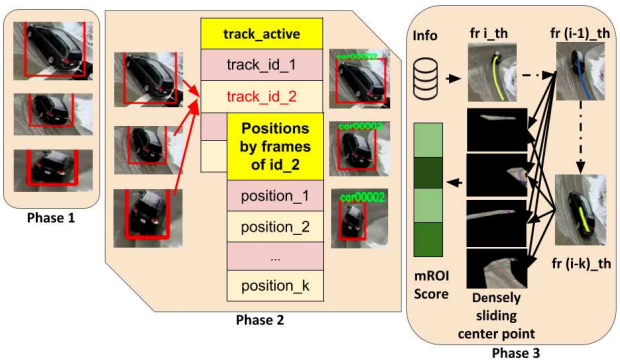

Figure 1. Overview of real-time vehicle flow counting task.

and truck. We use CenterNet [6] and alternatively switch between multiple backbones, including ResNet-34, ResNet-50, ResNet-101 [7], and DLA-34 [47], to evaluate both accuracy and computational efficiency of our detectors. We also consider tiny vehicle detectors [40] to handle vehicles that are far away from the traffic camera. However, thanks to the reliable m-ROIs (presented in Section 3.2), we can ignore tiny vehicle instances when counting vehicles.

To quickly associate vehicles across frames, we employ a simple yet efficient tracking method based on IoU (see Figure 2). To reduce processing time, we detect and track vehicles across frames with a skip-frame strategy. We define n_{step} as the interval between two consecutive frames that we process ($n_{step} = 1, 2, 3, etc$). As we do not process all frames, the bounding boxes of the same vehicle in two consecutive processed frames may not be overlapped enough comparing to considering all frames, especially when the vehicle moves fast. Thus, we propose to expand the bounding boxes of detected vehicles to match its bounding boxed across selected frames. Through experiments, we decide to use $n_{step} = 2$, which means that we can drop 50% frames and reduce half processing time while maintaining acceptable accuracy.

3.2. Motion-specific Region-of-Interest for Reliable Vehicle Tracking and Counting

One essential task in our proposed solution is to define a collection of motion-specific ROIs (m-ROIs); each corresponds to a given motion-of-interest (MOI). Figure 2 illustrates the process of creating a good set of m-ROIs from a given set of MOIs. We first extract the initial raw m-ROIs based on the annotated screenshot of each video given by organizers. An initial m-ROI can be determined as the whole area that can be crossed by a vehicle moving along the corresponding MOI. Our manually extracted m-ROI are processed into binary masks for each motion flow and can be overlapped with each other.

The set of raw m-ROIs extracted above, however, is not guaranteed to be a good choice for vehicle tracking and counting in each motion flow. Therefore, we define the two main criteria to refine for a good set of m-ROIs as follows:

- Minimize obstacles for vehicle tracking in an m-ROI, such as shadow or occlusion by traffic signs or trees. In Figure 2, the m-ROI$_3$ should avoid the area covered with trees after a vehicle turns right as it would easily lose track of vehicles at this area.
- Maximize the separation between different m-ROIs to reduce the possibility of confusing vehicles in different motion flows. In Figure 2, the m-ROI$_2$ (green) and m-ROI$_3$ (purple) should avoid going too close together to prevent possible confusion.

After defining a good set of MOIs for each camera location, we can now count vehicles moving along a specific

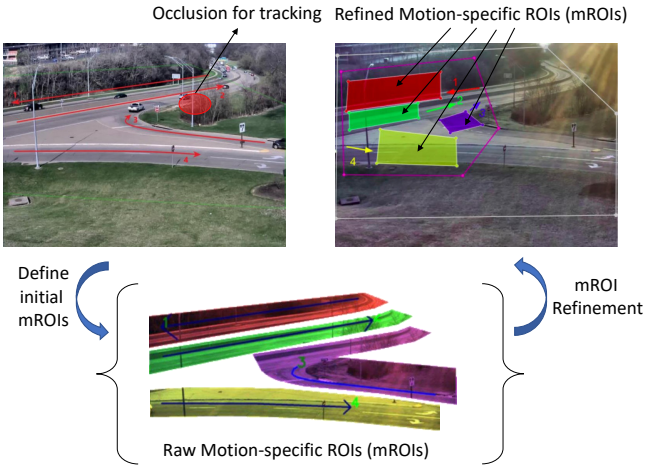

Figure 2. From motion-of-interest (MOI) to motion-specific ROI (m-ROI).

MOI. To determine which motion flow a vehicle is following, we exploit the two main properties of a motion-of-intest: its direction and refined m-ROI. The direction of a MOI is represented as a sequence of points along the motion direction. Both densely appearing attributes and directional information of vector points are valuable for determining the motion of objects. To reduce the impact of losing vehicle tracking, we determine the vehicle belonging to a motion flow by first counting the number of times it is in the refined m-ROI, then further verifying its trajectory against the motion direction.

We use two strategies to assign a vehicle to a specific region of a motion-of-interest. Our first strategy is based on the projected center of the bounding box on all binary masks of m-ROIs. By densely sliding through multiple m-ROIs, we can attach each detected object with the MOI that has the most occurrence of object' center points on its mask region m-ROI (see Figure 3.2).

Our second solution to find the m-ROI to which each vehicle belongs is to consider the path, created by connecting the centers of bounding boxes, not as a path, but as a region (c.f. Figure 3.2). We do this by expanding the path perpendicularly. For each extracted region and an m-ROI, we compute the overlapped area S_O, and the area within the region but outside the m-ROI, denoted as S_L. These two values are then used to compute the "compatibility" between an extracted region and an m-ROI. More specifically, the compatibility score is defined as $S_O - n_{penalty} \times S_L$. We multiply S_L by $n_{penalty}$ to penalise m-ROI that has too much non-overlapped area with the current region. Low compatibility scores are ignored. For each region created from the trajectory of a vehicle, the most compatible m-ROI is selected.

When a vehicle goes out of the region m-ROI$_i$ of the i^{th} specific motion flow, there can be two scenarios: the

Figure 3. Motion paths generated from the centers of vehicles.

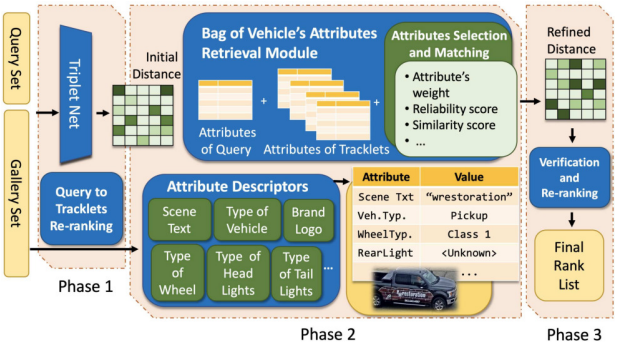

Figure 4. Our approach for Vehicle re-identification task with three phases.

vehicle also goes out the global region-of-interest, or it is still moving toward the exit edge in the ROI. In the former case, we simply increase the corresponding vehicle counter. In the latter case, we assume that the vehicle continues to move along the current MOI_i without any anomaly event and we can forecast the time instant when that vehicle actually leaves the global ROI based on the average time span of all vehicles moving along that motion path MOI_i to finish the path toward the exit of the global ROI.

4. Multi-Camera Vehicle Re-Identification with Bags of Vehicle Attributes

4.1. Overview

Given two set of vehicle images: Query set $\mathcal{Q}$, and gallery set $\mathcal{G}$, the goal of vehicle re-identification task is retrieving a list of images $[g_1, g_2, ..., g_k | g_i \in \mathcal{G}]$ which have the same identity with a given query $q \in \mathcal{Q}$. Another useful information is each g is aligned to one and only one tracklet $T \in \mathcal{T}$. In our approach, we want to utilize the intra-tracklet variability by instead of matching a query image to all images in the gallery set individually, we do it on the tracklet level, and return belonging gallery images in order.

Our proposed solution consists of three main phases, with a novel vehicle- attribute-based retrieval component as illustrated in Figure 4. In the first phase, a deep metric embedding network is trained to learn a function $f(x)$to map an input image x_i to its latent representation $\mathbf{f}_i$ with $\mathbf{f}_i \in \mathbb{R}^D$, and D is the dimension of the embedded vector. Simply stated, if $\mathbf{f}_i$ and $\mathbf{f}_j$ belong to the same instance, our goal is minimizing the distance $\mathcal{D}_{feat}(\mathbf{f}_i, \mathbf{f}_j)$ between two vector and then get the initial distance between every q and T, called $\mathbf{D}_{init}$. However, despite the acceptable performance in overall shapes and vehicle colors, deep metric embedding seems to be failed to distinguish between two vehicles base on their specific details, such as their type of wheels, headlights, unique textures, etc. To tackle with the mentioned problem, we proposed a set of vehicle attributes $\mathcal{A}$ where $\mathcal{A}_k$ is an attribute with its set of possible values $\mathcal{A}_k^D$ that we believe is a piece of useful information for re-identification. Given an image x, it is now represented as a pair of (a, c) where $a^k \in \mathcal{A}_k^D, c^k \in \mathbb{R}$ are value and confident score of the k^{th} attribute, respectively. Then, we define function $\mathcal{D}_{attrb}(q, T)$ to measure the similarity between attributes of a query and tracklets, forming a new distance matrix $\mathbf{D}_{attr}$.

The final distance $\mathbf{D}_{refined}$ is calculated based on the two mentioned matrices, follow up by some re-ranking and result verification approaches to get the final rank list.

4.2. Initial Distance by Deep Metric Embedding

As a baseline, Triplet Net [9] is used as our main deep metric embedding architecture with Efficient Net [37] backbone $f(x)$, an input image is embedded into a 1280 dimensions feature vector. Distance $\mathcal{D}_{feat}$ is Euclidean Distance. In addition, the Online Triplet Loss [31] with a hard margin equal 1.0 and Batch Hard sampling [8] with $p = 50$ and $k = 3$ is chosen. For each tracklet T, represent vector is assigned by $\mathbf{t} = \text{average} f(g), \forall g \in T$, among their vector dimensions. Similarity between a query image q and T is calculated as $\mathcal{D}_{feat}(f(q), \mathbf{t})$. We also applied the work from [50] to have the initial distance matrix $\mathbf{D}_{init}$ with its element $d_{i,j}$ is the re-ranked distance between q_i and T_j.

4.3. Enhancing Re-Identification with Bag of Vehicle's Attribute Retrieval

Inspired by Bag of Words in natural language processing, in this scenario, a tracklet consists of several images can be treated as a paragraph contains a number of words. To recall, we pre-defined a set of attributes $\mathcal{A}$ contains some effective attributes for the re-identification problem. An associating set of attribute descriptors $\Phi(x)$ can extract the attributes information, called a with the corresponding confident score c of a given image x. Note that we do not want to ruin $\mathbf{D}_{init}$, new attributes need to be qualified before being used. A set of threshold values for confident scores Λ is

Figure 5. Samples with the same value in some selected attributes set. In test time, those attributes are detected automatically.

used to filter out weak attributes. The contribution of each attribute to the attribute distance is weighted by a set of γ values. Remarkably, in the previous stage, two images are compared by their visual information, now the similarity is measured by the two sets of reliable attributes only.

Constructing attributes set $\mathcal{A}$. In this prior work, $\mathcal{A}$ includes: $\mathcal{A}_1$: scene text (text), $\mathcal{A}_2$: vehicle-type (6 classes), $\mathcal{A}_3$: fine-grain vehicle type (8 classes), $\mathcal{A}_4$: wheel type (6 classes), $\mathcal{A}_5$: camera view point (7 classes), $\mathcal{A}_6$: tail light (7 classes), $\mathcal{A}_7$: vehicle roof (3 classes) and $\mathcal{A}_8$: wheel similarity in low level features ($\mathbb{R}$). Selected examples of $\mathcal{A}$ is given in Figure 5. Manual annotations for those attributes on training set are available online for future research. Proposing additional annotations for other attributes, are also potential future extensions.

Attribute descriptors Φ. Noticeably, Φ is different between each attribute. For instance, scene text attribute $\mathcal{A}_1$: is obtained by using Φ_1 from [2, 3], while Φ_{2-7} : follow a same pipeline. With a given image, Faster-R-CNN [29] is used to crop out all regions of interest. Each region is then going through a simple classifier with ResNet [7] backbone. Labels and confidence scores are returned afterward. Spectacularly, since descriptors are independent, we can enhance the robustness by stacking additional descriptors, utilizing intermediate results of previous ones. $\mathcal{A}_8$ is an example.

The vehicle wheel is one of the richest attributes to distinguish between two vehicles. However, comparing to $\mathcal{A}_4$, traditional approaches for embedding wheels seem to be more effective. Φ_8 is a measurement of similarity between two patches of the wheel, using only low-level features. Taking all cropped patches from $\mathcal{A}_4$, SIFT is used to construct keypoint descriptors for each image, their histogram is used as an embedding vector. With a large number of patches, we filter out ones that have the number of keypoints smaller than a given threshold. To make the comparison, L2 distance between two corresponding vectors can be calculated easily. Demonstrating in Figure 13, given an image, $\mathcal{A}_8$ can point out a list of other images that cannot share the same identity. The confidence score from Φ_8 is calculated base on the associating level between an image with others, group by their associating tracklets.

Significantly, the variety of Φ is not limited to any specific categories, in this work, Φ can be an image classifier, scene text detector with deep learning approaches, and now it is a low-level features similarity criterion. There are also plenty of ways we can use attribute values, they can bring two images together ($\mathcal{A}_{1-7}$) or separating them out ($\mathcal{A}_8$).

Bag of vehicle's attributes retrieval. The goal of this step is establishing the $\mathbf{D}_{refined}$ distance matrix by using new vehicle attributes profile for each image. Let denote $a^k, c^k = \Phi_k(x)$ with $x \in \mathcal{Q} \cup \mathcal{G}$ and λ_k is the threshold for the k^{th} attributes in $\mathcal{A}$. For each a^k, new value is assigned by:

$$a^k = \begin{cases} a^k & \text{if } c^k \geq \lambda_k \\ \varnothing & \text{else} \end{cases} \tag{1}$$

By using the new assigned value, suppose the k^{th} attribute of a query image q_i has value a_i^k, that value in the j^{th} tracklet is given by $\alpha_j^k = \text{mode}[a_u^k | x_u \in T_j, a_u^k \neq \varnothing]$ with mode[.] returns the value that appears most often. In sort, the majority voting among images in the same tracklet is performed, after filtering out $\varnothing$ values. The distance between q_i and T_j is calculated by:

$$\mathcal{D}_{attr}(q_i, T_j) = -\sum_{k}^{|\mathcal{A}|} \mathbb{I}(a_i^k = \alpha_j^k).\gamma_k \tag{2}$$

where $\mathbb{I}(e)$ is an indicator function, $\mathbb{I}(e) = 1$ if e is true, and equal 0 otherwise. However, $\mathbb{I}(e)$ can be changed to become more flexible. With scene text $\mathcal{A}_1$, we still allow two strings have up to 3 different characters when matching. γ_k is the weight of the k^{th} attribute. The sign of γ_k depends on that attribute aims to reduce or increase the initial distance. The attribute distance matrix $\mathbf{D}_{attr}$ can be formed by using Eq.2 for all pairs of query images and tracklets.

4.4. Refined Distance and Finalizing

In general, Sections 4.2 and 4.3 bring us two distance matrices. Finally, the refined distance is given by $\mathbf{D}_{refined} = \alpha\mathbf{D}_{init} + \beta\mathbf{D}_{attr}$, where α and β are scaling factors. In our experiments, they are weighted equally. With the i^{th} query image q_i, tracklets associating with q_i is $\Pi_i = \text{argsort}\mathbf{D}_{refinned}[i, :]$. To create the final rank list R_i for the i^{th} query , we just need to return gallery images belong to those tracklets in order: $R_i = [g | g \in T_u, u \in \Pi_i]$.

5. Traffic Anomaly Detection with GAN-based Forward Prediction and Backward-Tracking Refinement

5.1. Overview

Figure 6 illustrates the overview of our solution for traffic anomaly detection with three main phases: video pre-processing, stalled vehicle detection with multiple adaptive detectors, and anomaly event refinement with forward and backward strategies.

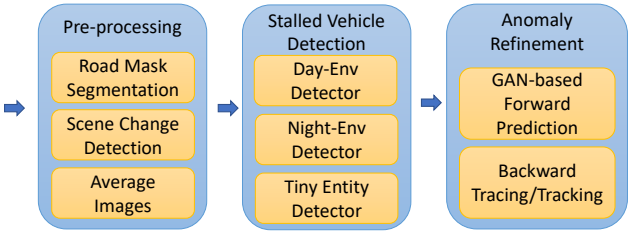

Figure 6. Overview of our proposed anomaly detection system.

The key idea of our work is that we aim to determine the time instant when an anomaly event happens accurately. However, there is no official definition for such events. In AI City 2020 Challenge, anomaly events mainly fall into two categories: stalled vehicles and crashes. The convention used is that in case of a stalled vehicle, the start time is the time when it comes to a complete stop. In the case of multiple crashes, the start time is the instant when the first crash occurs.

After detection, we utilize two different approaches to determine when the event actually started. Section 5.2 and Section 5.3 describe these two approaches in detail.

Figure 7 shows an example for anomaly detection. Most of the pre-processing techniques are adopted from our solution in AI City Challenge 2019 [28]. We first apply scene change detection with an LBP-based approach to split the video clip into multiple scenes in which the camera does not change perspective significantly. For each scene, we detect and skip frozen frames, the consecutive frames with

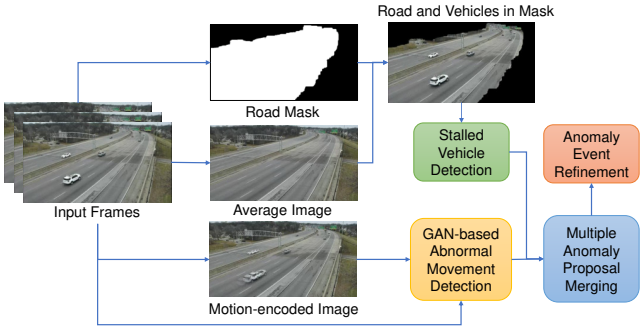

Figure 7. Example of anomaly detection process.

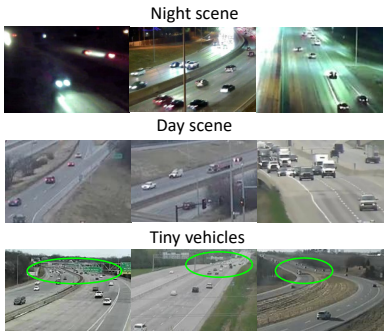

Figure 8. Multiple adaptive vehicle detectors for different contexts.

nearly identical content. Such frozen frames may lead to wrong stalled vehicle detection as a vehicle may appear for a long period, and may not provide sufficient motion information for the input of GAN-based future frame prediction (see Section 5.2). Then we calculate a road mask to focus only on regions of interest, i.e., main roads.

In [28], we use the background modeling method with average images to remove moving vehicles. In our current solution, this technique is used to generate two sequences of average images targeting two objectives: to remove moving vehicles and to encode motion information.

An average image avg_i is a calculated as weighted combination between the current frame $frame_i$ and its previous average image avg_{i-1}. The defined the coefficient α to represent the contribution of the current frame $frame_i$ in the average image avg_i. We use a small value ($\alpha = 0.01$) , similar to the work of Xu et.al[46], for moving object removal. A stalled vehicle becomes visible in an average image when it stops long enough.

The problem now is to determine when that vehicle begin to stop. This motivates our proposal for phase 3 for anomaly event refinement (see Section 5.2 and Section 5.3). For motion encoding, we use a larger value of α, such as 0.5, to blend recent frames into a single motion-encoded image.

For every anomaly event, there usually exists at least one stalled vehicle. From that observation, we mostly focus on improving stalled vehicle detection methods on low-resolution videos. Instead of using only a single vehicle detection model to handle all cases, we use multiple context-based models with high precision to improve results on each environment [41]. From the training set of Track 4 in AI City Challenge 2020, we prepare data for different contexts to train vehicle detectors (either Faster-RCNN or Centernet) for day and night scenes, and also for tiny vehicles. Figure 8 illustrates some example images in different contexts.

5.2. GAN-based Future Frame Prediction

In the case of car crashes, we note that there is usually an abrupt change in the trajectory of a vehicle before it crashes. We aim to predict the normal trajectory by using a GAN-based method to generate the next frame, then compare it with the actual next frame to see if any abnormal phenomenon occurs. Thus, we use our GAN-based future frame prediction for traffic surveillance videos [16] to generate the next frame for a given frame in a usual scenario, and check the generated image against the real next frame to detect a potential anomaly.

Figure 9 illustrates the overview of our proposed GAN-based method to detect an anomaly by checking a predicted future frame from a current frame and a motion-encoded information against the real next frame. If the difference is within a given threshold, we conclude that there is an anomaly event. Another property of this method is that it can also detect a vehicle moving abruptly, e.g., changing lane; however, this situation rarely occurs in the testing dataset of AI City Challenge 2019 and 2020.

Motion encoding with blending: Given a frame, we aim to generate the next frame and compare it with the actual frame to see if any abnormal phenomenon occurs. However, a single frame does not carry sufficient information to deduce the motion of an object. Instead of supplying k frames to input, we encode motion information by blending several consecutive frames into an average image, as described in Section 5.1. As we want to preserve moving vehicles and also their trajectory, we use a larger $\alpha = 0.5$. This results in moving vehicles, leaving blurry trails on their path, serving as past information for motion prediction.

Loss functions in GAN training process: To train a multiscale UNet generator for future frame prediction, we use four loss functions,including L2 Loss (L2), Gradient Different Loss (GDL), Adversarial loss (Adv), and Optical Flow Loss (OFL). We further enhance the quality for the boundary areas of vehicles with Scaled Intensity Loss (SIL), proposed in [16]. Our purpose here is to increase the differences at the boundary of vehicles for GAN to enhance vehicle boundary areas.

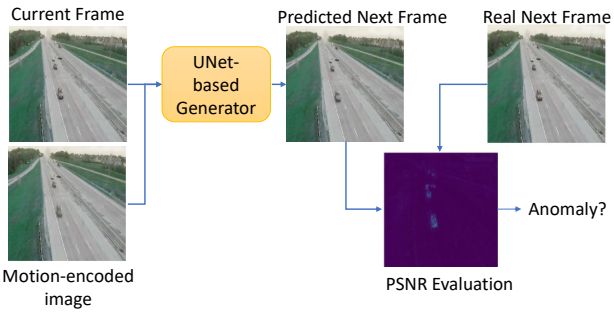

Figure 9. GAN-based future frame prediction with motion-encoded information for anomaly detection.

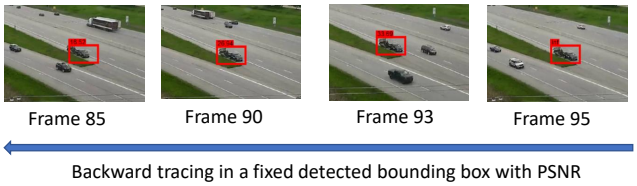

Figure 10. Backward tracing in a fixed detected bounding box to determine the moment when a vehicle stops.

Scoring of an anomaly event: We use the Peak Signal-to-Noise Ratio (PSNR) [24] to calculate the likelihood of two frames. A higher value of PSNR means the pair of images are similar. If PSNR falls below a certain threshold, an anomaly is likely to happen.

5.3. Backward Vehicle Tracing/Tracking

When detecting anomalies in an average image, we observe that an anomaly event has happened a few seconds before we can discover it. This is because a stalled vehicle takes time to contribute enough to the average image to be detected. Using this idea, from the detected vehicle, we trace back on the original frames of the video to find the exact instant when the vehicle stops. We go backward in time until the local region defined by the detected bounding box becomes too different from the detected vehicle. Figure 10 shows an example for backward tracing of a stalled vehicle and its previous frames. We also use a fast tracking method [43] to backward track the trajectory of a stalled vehicle to identify when it begins to change lane before stopping (see Figure 14) .

6. Experimental Results

In this section, we briefly report our results on the three datasets of Track 1, Track 2, and Track 4 in AI City Challenge 2020.

Track 1: Vehicle Flow Counting by Class

Table 1 shows the final ranking of Track 1. Our method achieves the 10^{th} place among 18 team submissions with the S1 score of 0.8297. Figure 11 shows some examples of

Table 1. Ranking result on Track 1

Rank	Team ID	Team Name	S1 Score
1	99	Everest	0.9389
2	110	CSAI	0.9346
3	92	INF	0.9292
4	60	DiDiMapVision	0.9260
5	20	6thAI	0.9236
...	...	...	...
10	**80**	**HCMUS**	**0.8297**
11	119	PES	0.8254
12	108	Traffic_Flaw_Theory	0.8138
...	...	...	...

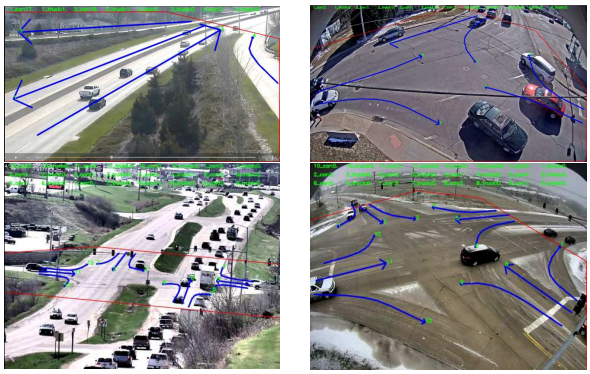

Figure 11. Visualization of our results in Track 1.

our results. By defining disjoint motion-specific ROIs (m-ROIs), we can improve the accuracy for counting vehicles in different MOIs that are close to each other, especially in intersections, and achieve the Effectiveness score of 0.8011. By skipping 50% frames, we can speed up the processing time and our method has the Efficiency score of 0.8477.

Track 2: Vehicle Re-identification

Table 2 shows the mAP score of our method in the vehicle re-identification dataset of Track 2 in AI City Challenge. Our method achieves mAP of 0.3882 and takes the 26^{th} place among 41 participating teams.

Table 2. Ranking result on Track 2 - Public leaderboard

Rank	Team ID	Team Name	mAP Score
1	73	Baidu-UTS	0.8413
2	42	RuiYanAI	0.7810
3	109	DMT	0.7322
4	26	IOSB-VeRi	0.6899
5	39	BestImage	0.6684
...	...	...	...
26	**80**	**HCMUS**	**0.3882**
...	...	...	...

Scene text is one of our selective attributes which shows a spectacular and explainable way to perform vehicle re-identification challenge. Some sample results are given in Figure 12. For buses with similar color and shape, bus numbers are an important clue for instance re-identification.

To illustrate the fine-grained matching, we demonstrate in Figure 13 the wheel matching result with Bag of Features (Green box: template patches, yellow box: candidate patches. Red box: underqualified matching tracklets).

Figure 12. Scene text is used to match two given vehicle images.

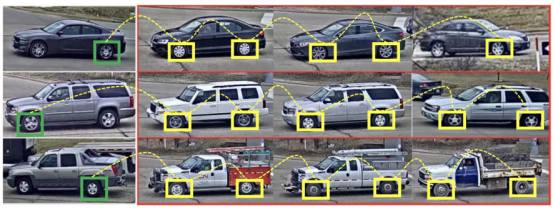

Figure 13. Wheel matching with Bag of Features ($\mathcal{A}_8$).

Track 4: Anomaly Detection We obtain the 5^{th} place out of 13 team submissions. The final ranking of Track 4 is showed in Table 3. In final result, we achieve F1 score of 0.9412, RMSE of 11.2556, and S4 Score of 0.9059.

Table 3. Ranking result on Track 4

Rank	Team ID	Team Name	S3 Score
1	113	Firefly	0.9695
2	114	stu	0.9615
3	51	SIS Lab	0.9494
4	75	Albany_NCCU	0.9494
5	**80**	**HCMUS**	**0.9059**
6	109	cet	0.6194
...	...	...	...

In row 1 of Figure 14, we illustrate the quality of a generated frame with a real frame in a regular scenario. This technique can be used to predict an anomaly in future frames[16], or can be used to refine the moment an abnormal event begins to occur. Our method can also backward tracking from a stalled vehicle to find its past trajectory and the moment when it begins to move in a sudden-changed path (see row 2 in Figure 14).

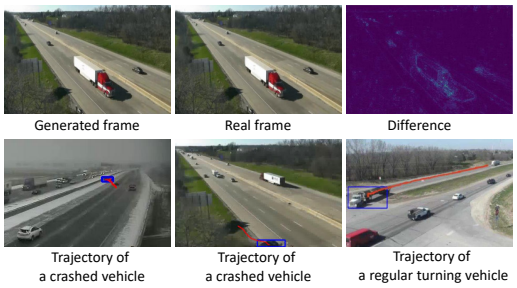

Figure 14. Examples for frame prediction and backward tracking.

7. Conclusion

We introduce an Intelligent Traffic Analysis Software Kit (iTASK) to tackle challenging problems of traffic video analysis, including vehicle flow counting, vehicle re-identification, and anomaly detection. In 3 years participating in AI City Challenge , we gradually develop different components for multiple traffic analysis tasks, and our library is designed as an open environment to add more algorithms and components to enhance the results and also to handle more challenging tasks.

Acknowledgements

This research is supported by Vingroup Innovation Foundation (VINIF) in project code VINIF.2019.DA19. We would like to thank AIOZ Pte Ltd for supporting our research team with computing infrastructure.

References

[1] X. Z. A. Kanaci and S. Gong. Vehicle re-identification by fine-grained cross-level deep learning. In *5th Activity Monitoring by Multiple Distributed Sensing Workshop, British Machine Vision Conference*, pages 1–6, July 2017.

[2] J. Baek, G. Kim, J. Lee, S. Park, D. Han, S. Yun, S. J. Oh, and H. Lee. What is wrong with scene text recognition model comparisons? dataset and model analysis. In *International Conference on Computer Vision*, 2019.

[3] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee. Character region awareness for text detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 9365–9374, 2019.

[4] H. Chen, B. Lagadec, and F. Bremond. Partition and reunion: A two-branch neural network for vehicle re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[5] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik. A real-time computer vision system for vehicle tracking and traffic surveillance. *Transportation Research Part C: Emerging Technologies*, 6(4):271–288, 1998.

[6] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian. Centernet: Keypoint triplets for object detection. *International Conference on Computer Vision*, 2019.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[8] A. Hermans, L. Beyer, and B. Leibe. In defense of the triplet loss for person re-identification. *CoRR*, abs/1703.07737, 2017.

[9] E. Hoffer and N. Ailon. Deep metric learning using triplet network. In *SIMBAD*, 2014.

[10] J.-W. Hsieh, S.-H. Yu, Y.-S. Chen, and W.-F. Hu. Automatic traffic surveillance system for vehicle tracking and classification. *Trans. Intell. Transport. Sys.*, 7(2):175–187, Sept. 2006.

[11] P. Huang, R. Huang, J. Huang, R. Yangchen, Z. He, X. Li, and J. Chen. Deep feature fusion with multiple granularity for vehicle re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[12] T.-W. Huang, J. Cai, H. Yang, H.-M. Hsu, and J.-N. Hwang. Multi-view vehicle re-identification using temporal attention model and metadata re-ranking. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[13] H. Jung, M.-K. Choi, J. Jung, J.-H. Lee, S. Kwon, and W. Young Jung. Resnet-based vehicle classification and localization in traffic surveillance systems. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, July 2017.

[14] M. Kampelmühler, M. G. Müller, and C. Feichtenhofer. Camera-based vehicle velocity estimation from monocular video. *Computer Vision Winter Workshop*, 2018.

[15] A. Kanaci, M. Li, S. Gong, and G. Rajamanoharan. Multi-task mutual learning for vehicle re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[16] N. Khac-Tuan, D. Dat-Thanh, D. Minh N., and M.-T. Tran. Anomaly detection in traffic surveillance videos with gan-based future frame prediction. In *Proceedings of the 2020 International Conference on Multimedia Retrieval ICMR 2020*, 2020.

[17] P. Khorramshahi, N. Peri, A. Kumar, A. Shah, and R. Chellappa. Attention driven vehicle re-identification and unsupervised anomaly detection for traffic understanding. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[18] P.-K. Kim and K.-T. Lim. Vehicle type classification using bagging and convolutional neural network on multi view surveillance image. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, July 2017.

[19] T.-N. Le, A. Sugimoto, S. Ono, and H. Kawasaki. Attention r-cnn for accident detection. In *IEEE Intelligent Vehicles Symposium*, 2020.

[20] B. Li, T. Wu, C. Xiong, and S.-C. Zhu. Recognizing car fluents from video. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2016.

[21] C.-T. Liu, M.-Y. Lee, C.-W. Wu, B.-Y. Chen, T.-S. Chen, Y.-T. Hsu, and S.-Y. Chien. Supervised joint domain learning for vehicle re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[22] X. Liu, W. Liu, H. Ma, and H. Fu. Large-scale vehicle re-identification in urban surveillance videos. In *IEEE International Conference on Multimedia and Expo*, pages 1–6, July 2016.

[23] K. Marotirao Biradar, A. Gupta, M. Mandal, and S. Kumar Vipparthi. Challenges in time-stamp aware anomaly detection in traffic videos. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[24] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. *International Conference on Learning Representations*, 2016.

[25] M. Naphade, M.-C. Chang, A. Sharma, D. C. Anastasiu, V. Jagarlamudi, P. Chakraborty, T. Huang, S. Wang, M.-Y. Liu, R. Chellappa, J.-N. Hwang, and S. Lyu. The 2018 nvidia ai city challenge. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2018.

[26] M. Naphade, Z. Tang, M.-C. Chang, D. C. Anastasiu, A. Sharma, R. Chellappa, S. Wang, P. Chakraborty, T. Huang, J.-N. Hwang, and S. Lyu. The 2019 ai city challenge. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[27] K.-T. Nguyen, D.-T. Dinh, M. N. Do, and M.-T. Tran. Anomaly detection in traffic surveillance videos with gan-based future frame prediction. In *International Conference on Multimedia Retrieval*, 2020.

[28] K.-T. Nguyen, T.-H. Hoang, M.-T. Tran, T.-N. Le, N.-M. Bui, T.-L. Do, V.-K. Vo-Ho, Q.-A. Luong, M.-K. Tran, T.-A. Nguyen, T.-D. Truong, V.-T. Nguyen, and M. N. Do. Vehicle re-identification with learned representation and spatial verification and abnormality detection with multi-adaptive vehicle detectors for traffic video analysis. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[29] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, 2015.

[30] M. Riveiro, M. Lebram, and M. Elmer. Anomaly detection for road traffic: A visual analytics framework. *IEEE Transactions on Intelligent Transportation Systems*, 18(8):2260–2270, Aug 2017.

[31] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[32] Y. Shen, T. Xiao, H. Li, S. Yi, and X. Wang. Learning deep neural networks for vehicle re-id with visual-spatio-temporal path proposals. In *International Conference on Computer Vision*, pages 1918–1927, Oct 2017.

[33] H. Shi. Geometry-aware traffic flow analysis by detection and tracking. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2018.

[34] N. Silva, J. Soares, V. Shah, M. Y. Santos, and H. Rodrigues. Anomaly detection in roads with a data mining approach. *Procedia Comput. Sci.*, 121(C):415–422, Jan. 2017.

[35] J. Sochor, J. Špaňhel, and A. Herout. Boxcars: Improving fine-grained recognition of vehicles using 3-d bounding boxes in traffic surveillance. *IEEE Transactions on Intelligent Transportation Systems*, PP(99):1–12, 2018.

[36] J. Spanhel, V. Bartl, R. Juranek, and A. Herout. Vehicle re-identifiation and multi-camera tracking in challenging city-scale environment. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[37] M. Tan and Q. V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *International Conference on Machine Learning*, 2019.

[38] X. Tan, Z. Wang, M. Jiang, X. Yang, J. Wang, Y. Gao, X. Su, X. Ye, Y. Yuan, D. He, S. Wen, and E. Ding. Multi-camera vehicle tracking and re-identification based on visual and spatial-temporal features. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[39] Z. Tang, M. Naphade, M.-Y. Liu, X. Yang, S. Birchfield, S. Wang, R. Kumar, D. Anastasiu, and J.-N. Hwang. CityFlow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 8797–8806, 2019.

[40] M.-T. Tran, T. Dinh-Duy, T.-D. Truong, V. Ton-That, T.-N. Do, Q.-A. Luong, T.-A. Nguyen, V.-T. Nguyen, and M. N. Do. Traffic flow analysis with multiple adaptive vehicle detectors and velocity estimation with landmark-based scanlines. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2018.

[41] M.-T. Tran, T. Dinh-Duy, T.-D. Truong, V. Ton-That, T.-N. Do, Q.-A. Luong, T.-A. Nguyen, V.-T. Nguyen, and M. N. Do. Traffic flow analysis with multiple adaptive vehicle detectors and velocity estimation with landmark-based scanlines. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 100–107, 2018.

[42] G. Wang, X. Yuan, A. Zheng, H.-M. Hsu, and J.-N. Hwang. Anomaly candidate identification and starting time estimation of vehicles from traffic videos. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[43] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. Torr. Fast online object tracking and segmentation: A unifying approach. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1328–1338, 2019.

[44] T. Wang, X. He, S. Su, and Y. Guan. Efficient scene layout aware object detection for traffic surveillance. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, July 2017.

[45] M. Wu, G. Zhang, N. Bi, L. Xie, Y. Hu, and Z. Shi. Multi-view vehicle tracking by graph matching model. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[46] Y. Xu, X. Ouyang, Y. Cheng, S. Yu, L. Xiong, C.-C. Ng, S. Pranata, S. Shen, and J. Xing. Dual-mode vehicle motion pattern learning for high performance road traffic anomaly detection. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2018.

[47] F. Yu, D. Wang, and T. Darrell. Deep layer aggregation. *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[48] J. Zhao, Z. Yi, S. Pan, Y. Zhao, Z. Zhao, F. Su, and B. Zhuang. Unsupervised traffic anomaly detection using trajectories. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[49] Z. Zheng, T. Ruan, Y. Wei, and Y. Yang. Vehiclenet: Learning robust feature representation for vehicle re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.

[50] Z. Zhong, L. Zheng, D. Cao, and S. Li. Re-ranking person re-identification with k-reciprocal encoding. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.