Continual learning-based Video Object Segmentation

by

Amir Nazemi

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2023

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Minglun Gong Professor, School of Computer Science, University of Guelph
Supervisor:	Paul Fieguth Professor, Dept. of Systems Design Engineering, University of Waterloo
Internal Member:	Bryan Tripp Professor, Dept. of Systems Design Engineering, University of Waterloo
	Mohammad Javad Shafiee Research Assistant Professor, Dept. of Systems Design Engineering, University of Waterloo
Internal-External Member:	Yuri Boykov Professor, Cheriton School of Computer Science, University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

The papers listed below are used in this thesis. I co-authored the papers and made significant contributions to the material's design, development, assessment, and writing.

Tong J*, Nazemi A*, Shafiee M, Fieguth P. CLVision 2020 Challenge Solution. Selected as one of the continual learning challenge finalists.
https://sites.google.com/view/clvision2020/challenge/challenge-winners.
This manuscript is incorporated in Chapter 1, and 2 of this thesis.

Nazemi A, Moustafa Z, Fieguth P. CLVOS23: A Long Video Object Segmentation Dataset for Continual Learning. arXiv preprint arXiv:2304.04259. 2023 Apr 9. Accepted in CVPR CLVISION workshop, 2023.
This paper is incorporated in Chapter 2, 3, and 4 of this thesis.

Nazemi A, Shafiee M, Gharaee Z, Fieguth P. Memory-Efficient Continual Learning Object Segmentation for Long Videos. Submitted to IJCV, 2022 Dec 16.
This paper is incorporated in Chapter 2, 3, 4, 5, 6, and 7 of this thesis.

Abstract

Machine learning models, specifically deep convolutional neural networks, have exceeded human-level performance in many research areas, such as object classification and voice recognition. However, they are not comparable to humans in real-world learning scenarios when the training data is non-i.i.d. infinite streaming data. An example of those real-world scenarios is continual learning.

Continual learning, as a new area of research in the field of machine learning, has become quite popular. It is the process of learning sequential data that comprises different domains and tasks. The main feature of a continual leaning problem is that the learning model does not have access to previously trained data. The main challenge of training a machine learning model on sequential data is catastrophic forgetting, which happens when a model forgets the previously learned tasks after being trained on new ones.

There are three different solutions for the problems of continual learning: prior-focused (regularization-based) solutions, likelihood-focused (rehearsal-based) solutions, and hybrid (ensemble) approaches.

In this thesis, semi-supervised video object segmentation (VOS) is addressed as a continual learning problem specifically for long video sequences, and three solutions are proposed. The first solution is Gated-Regularizer Continual Learning (GRCL) which is a prior-focused solution. The second proposed solution is aligned with likelihood-focused solutions and is Reconstruction-based Memory Selection Continual Learning (RMSCL). The third proposed solution is a hybrid solution (Hybrid) that benefits from GRCL and RMSCL.

All of the proposed solutions improve the performance of two baseline Online VOS methods (LWL and JOINT) but they can augment any online VOS and improve its performance on long videos.

Acknowledgements

I would like to express my deepest appreciation to my supervisor, Professor Paul Fieguth, for his unwavering support and invaluable guidance throughout my journey as a scientist and researcher. His knowledge, mentorship, and commitment have been pivotal in my academic and professional development.

I am indebted to my PhD committee members, Professors Tripp and Boykov, for their insightful comments, insightful suggestions, and the time they spent reviewing my work. I would like to express my appreciation to Dr. Shafiee, who has been my academic mentor for more than a decade. I am also extremely grateful to Professor Minglun Gong for taking the time to evaluate my thesis. I appreciate Dr. Zohreh Azimifar's encouragement and support for this exceptional PhD position at the University of Waterloo. Her confidence in my abilities and direction has been incredibly inspiring.

The members of the Vision and Image Processing Research Group at the University of Waterloo have my gratitude. Their camaraderie, support, and collaborative spirit have created a stimulating and productive environment in the laboratory. Lastly, I would like to express my gratitude to everyone who has helped realize this thesis. Your involvement, no matter how big or small, has made a significant difference in my academic journey, and I am truly grateful.

I am also grateful to Microsoft Office Media Group, NSERC Alliance, and the Digital Research Alliance of Canada (alliancecan.ca) for their generous support of this research.

Dedication

To my mom and dad, who have bravely thrown me into the world of Being and supported me every step of the way, I am forever grateful.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The understanding of the world that children have at the beginning of their lives is gradually expanded and refined throughout the course of their entire lives. Similarly, if we want to achieve the ultimate goal of artificial intelligence, we will need to construct intelligent models that interact with the world, learn like children, and continually update themselves on real-world events. In the field of machine learning, an evaluation scenario is more analogous to an assessment that would be carried out in a traditional educational environment. To make this point clear, an example of the evaluation of students in a school is provided. Usually, students have access to all of the materials that they require for a course exam, and they are given instructions on how to study and learn those resources for answering all of the questions prior to the exam. There is a problem with this learning and testing scenario for machine learning models. To explain the problem, a more specific example is provided in which an exam designed for preschoolers is used to classify different kinds of animals. If a young child already knows how to categorize the many different kinds of animals that exist, then informing that child about the many different kinds of trees that are out there should not cause that child to forget how to categorize animals. However, the mentioned example addresses a problem in machine learning that is referred to as catastrophic forgetting [2, 3, 4, 5]. Catastrophic forgetting is one of the primary obstacles that must be overcome in order to achieve continual learning. If we think of a child as an intelligent agent in the world, we can see that they are always learning new things, updating their knowledge, and judging how well they are doing based on a variety of inputs and provided labels. Thus, their training is not limited to the school's classes and academic evaluation. In other words, a real-world learning process for a child is an online continual learning process, and achieving this level of intelligence that can do online continual learning could be one of the most important goals of machine learning.

Continual learning is the process of learning a sequence of data consisting of different tasks and domains. Continual learning is also addressed as lifelong learning, sequential learning, and incremental learning [6, 7, 8]. In the general problem of continual learning, data come from an instance domain $\mathcal{X}$; however, the domain $\mathcal{X}$ can be changed over time, and this change can be gradual or abrupt. To refer to the school class example, children could have many classes in one day, and different classes could have different topics (different domains), and in each class, a student could also face different information with gradual changes in sub-topics.

In continual learning, it is important to handle the so-called abrupt change on the input domain, which is similar to the concept drift [9] on the input domain.

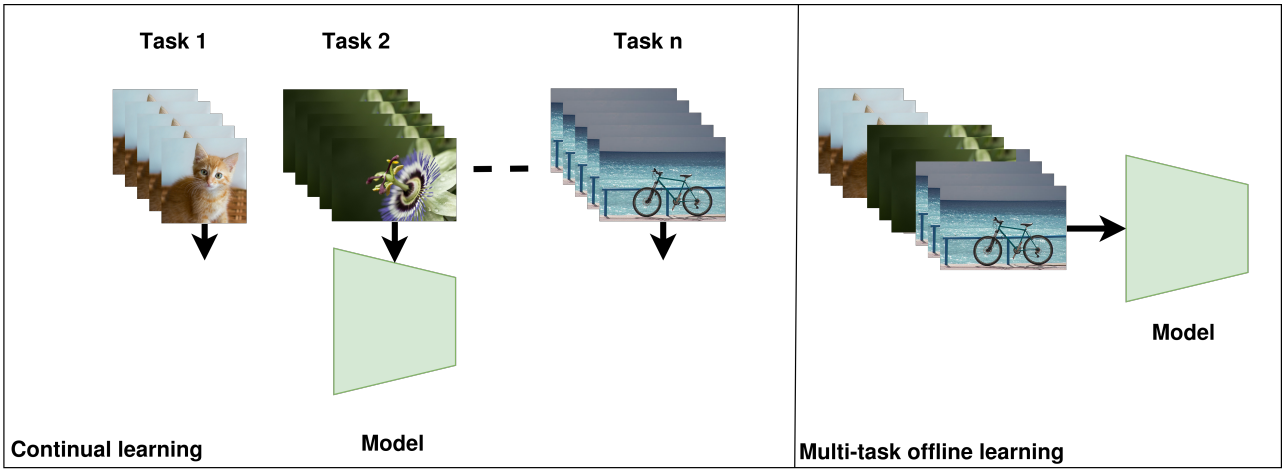

Figure 1.1: The continual learning and multi-task offline learning. In continual learning a model is trained on each task separately. However, in multi-task offline learning, the data for all tasks is available for the model at the same time.

Figure 1.1 shows the training scenario of a continual learning problem and a multi-task offline learning problem. In the training phase of the multi-task offline learning problem, the model has access to the data samples of all tasks, whereas in the continual learning scenario, the model has access to the sample data of only one task at a time. For example, the model is first trained on the "cats" category and does not have access to other categories. After training the model on the "cats" category, the model will be trained on the next category of data, which is "flowers, and during this training, the model will have no access to the "cats" category. It is worth noting that the independent and identically distributed (i.i.d.) assumption is not valid in continual learning [6]. The reason for this is

that in real world problems, there is no guarantee that samples are drawn independently from a distribution without distribution drift. In other words, The i.i.d. assumption implies an equal probability of seeing the current data in comparison to any other possible data where the current data are independent of past data. In practice, if the training data were shuffled, then the training data would be independently distributed; however, streaming data in an online learning scenario usually cannot be shuffled, and there are some temporal dependencies in the data stream, so the i.i.d. assumption is not valid for many real world problems where the data cannot be shuffled for training.

The main challenge in continual learning is catastrophic forgetting. The term catastrophic forgetting was first defined for Neural Networks [2, 3]; however, it could be a problem of other machine learning models [10]. It happens when a machine learning model is trained on a sequence of tasks while only having access to the training data of the current task and not the training data of preceding tasks. Consequently, the model updates its parameters (all of the parameters are changed) using the current task data and will forget some aspects of the proceeding tasks. For example, in Figure 1.1 the model is not able to classify "cats" well after training on "flowers". In an online-continual learning problem, the model should be trained online on streaming data, where task boundaries are not provided in the training. Thus, the model has to determine them by detecting concept drifts in the data.

In this thesis, Convolutional Neural Networks (CNNs) are considered the online learning model. CNNs are a type of neural network that is generally used in deep learning for image and video recognition, classification, and processing tasks. CNNs are made up of multiple layers of neurons that perform convolution. The convolution layer detects features such as edges, corners, and shapes in the input image by applying a set of filters, or kernels. Additionally, it will reduce the computational complexity of the training process using parameter sharing and local connectivity of the kernel operations [11] in comparison to fully connected layers in neural networks. CNNs are more effective when they become deeper, which means using more layers to build the network. When the number of layers is increased, the number of parameters (weights) in the network also increases, making it harder to train the networks well.

To address the challenges of training a deep CNN, a new set of Deep Convolutional Neural Networks (DCNNs) [12, 13, 14] was proposed that can be efficiently trained specifically on large datasets such as Imagenet [15]. DCNNs outperform other machine learning methods in many applications and areas; however, in continual learning scenarios, DCNNs suffer from the catastrophic forgetting problem. This problem is more visible in online-continual learning when the DCNNs have to be trained online on the streaming data. In this thesis, DCNNs are not going to be trained online; however, the baseline methods [16, 17] benefit

from trained DCNNs as feature extractors.

When we are talking about streaming data, the first thing that comes to mind is video. There are many fields of research in computer vision that deal with video data, such as activity recognition [18], object tracking [19], and Video Object Segmentation (VOS) [20].

This thesis studies and improves a subset of semi-supervised VOS approaches that deal with non-i.i.d. data (video sequences). In semi-supervised VOS solutions, the VOS model always has access, even during the evaluation time, to information about only one frame (the ground truth) of each video sequence. When dealing with long video sequences, this given information may become invalid after a period of time. In other words, because of the distribution drift that happens in videos and because the appearance of the object can be considerably changed, the given first frame ground truth may no longer be a valid label for the target object. The most recent semi-supervised solutions [16, 17, 21] incorporate some information from previously evaluated frames in order to segment the object in the current frame. Online VOS methods [16, 22, 23] are one type of semi-supervised VOS. These methods continuously train a part of their model on the frames that have been evaluated in order to perform VOS on the current frame. For the purpose of this thesis, I am going to approach the formulation of VOS from the perspective of continual learning. Moreover, three continual learning solutions are proposed to improve the performance of Online VOS on long video sequences. To the best of our knowledge, this is the first time that continual learning has been addressed in VOS approaches.

1.1 Challenges

Although CNNs are highly desirable and useful in many multi-task offline applications, there are many challenges for CNNs in continual learning:

- **Catastrophic forgetting:** Training the model on the current task should not degrade the model's performance on previously learned tasks. Catastrophic forgetting is the main challenge of continual learning.
- **The i.i.d. assumption:** The i.i.d. is an assumption, which makes the problems easier to solve by considering an identical distribution for data without distribution drift where data are drawn from the distribution independently.
- **Online learning:** Another significant challenge to continual learning is online learning. CNNs are trained by looping over a batch of data using the gradient of a loss

function. The training process is designed for a batch learning process; however, a general problem of continual learning as explained in [6] should avoid offline batch training. In other words, in an online-continual learning problem, CNNs models have to be trained and updated online.

- **Learning with limited memory and computational resources:** A simple solution for continual learning is to create a new model for each task and keep it in memory, or use one model and keep all of the observed data in memory. However, with a growing memory and a growing number of tasks, this is not possible. In other words, we want to develop solutions that do not assume infinite storage size, which requires an unbounded system as a solution. Thus, having a solution that tolerates a limited amount of memory and computational resources is another challenge.

- **Real-world scenario:** Many of the current datasets, setups, and scenarios for problems of continual learning are not practical enough [24]. In other words, the datasets, setups, and scenarios are not adequately addressing real-world problems. For example in [25] authors forms MNIST [26] dataset as stream data which is not a real world scenario.

Some of the aforementioned challenges are addressed in continual learning [27]; however, there is still room for improvement.

In this thesis, it is shown that a semi-supervised VOS solution which deals with video data faces continual learning challenges, specifically if the VOS solution has a CNN module for online learning.

- **Forgetting:** Given the distribution drift that occurs in long videos, a VOS model could forget its learning on the segment of the video prior to distribution drift. I would not call this catastrophic forgetting because the distribution drift is not as dramatic as domain shift in the standard continual learning scenario of learning to identify new image classes, but it is still forgetting.

- **The i.i.d. assumption:** Generally, video data, specifically long video data, is not i.i.d. since it has some temporal correlation between frames and also could have some domain shifts if the camera and point of view are changed.

- **Online learning:** Online VOS [16, 17, 28] is a good example of online learning-based solutions, where a part of the model is trained online and updated throughout the video frames.

- **Learning with limited memory and computational resources:** State-of-the-art VOS solutions all benefit from memory to store information from past evaluated frames to use for segmenting the current frame.

- **Real-world scenario:** There are many meaningful applications for VOS, such as augmented reality, autonomous driving, and robotic.

A solution for Online VOS should address all of these significant challenges. The major distinction between VOS and other video processing is that in VOS we need to do estimation on every frame of the video, which necessitates online training to improve the performance of the model, making it plausible as a target for a continual learning-based solution. On the other hand, for other video processing fields of research such as Video Action Recognition [18], the estimation is on the whole video sequence, and usually there is no need to do online learning on every frame of video.

1.2 Contributions

The main contributions of this thesis are as follows:

- The general formulation of continual learning on online video object segmentation (Chapter 3) is the first contribution of this thesis. A general Online VOS structure is defined and formulated from a continual learning perspective alongside the limitations and motivations of the proposed formulation.

- A prior-focused continual learning-inspired solution is proposed in Chapter 4. The proposed gated regularization-based continual learning (GRCL) approach can augment and improve any Online VOS model.

- Another contribution of this thesis is to propose a reconstruction-based memory selection continual learning (RMSCL) method (Chapter 5) for Online VOS models. The proposed likelihood-focused approach is able to add to Online VOS and improve its speed and accuracy.

- The last contribution of this thesis is the proposed the Hybrid (Chapter 6) continual learning approach that is a smart fusion of RMSCL and GRCL. The objective of Hybrid method is to improve robustness and accuracy of Online VOS on long videos.

1.3 Thesis structure

Chapter 2 begins with an overview of convolutional neural networks, and a discussion of the challenges associated with both online and continual learning. This chapter continues its presentation of a literature overview of contemporary approaches for the segmentation of video objects in long video sequences. In Chapter 3, a general formulation of Online VOS is proposed from the standpoint of continual learning.

In Chapter 4, the gated regularization-based continual learning (GRCL) approach is presented, and in Chapter 5, an alternate solution to GRCL known as the likelihood-focused method (RMSCL) based on a memory selection methodology is discussed. GRCL is presented as a prior-focused solution.

In Chapter 6, the suggested Hybrid technique is discussed. This method incorporates aspects of both GRCL and RMSCL. To put it another way, the Hybrid technique is a smart fusion of the GRCL and the RMSCL.

In Chapter 7, we go into depth about the experimental findings of the strategies that were suggested using long video sequences. In addition, the ablation research of the suggested approaches is discussed in Chapter 7.

The thesis is brought to a close with Chapter 8, which provides a summary of the contributions made by the thesis as well as suggested study paths for further work.

Chapter 2

Background

Ideal machine learning model should outperform humans; however, in many areas there is still a significant gap between machine learning and human performance. Online learning and continual learning are two major challenges in many machine learning problems including deep learning. Convolutional Neural Networks (CNNs) are widely used models for a wide range of machine learning tasks, particularly video processing. Video is one of the most complicated data in the machine learning field because it is temporal, high-dimensional, and highly variable, and consequently video analysis tasks such as Video Object Segmentation (VOS) are difficult. In order to perform video object segmentation, this dissertation addresses the continual learning of small CNNs on long video sequences. Convolutional neural networks are discussed in this chapter, as are online, continual, online-continual learning, and video object segmentation.

2.1 Convolutional Neural Networks

A Neural Network (NN) [29, 30, 31] is a function f that, given a set of input training data $X = [x_1, x_2, ..., x_n]$ and training target data $Y = [y_1, y_2, ..., y_n]$, learns to map input X to the target Y. As illustrated in Figure 2.1, a NN consists of some layers f_i that together form the function f. Thus, f is defined as:

$$f(x) = f_n\Big(f_{n-1}\big(...\big(f_1(x)\big)\big)\Big) \tag{2.1}$$

where f_n is the n^{th} layer in the NN.

In theory, if each layer function of NN is a linear function, then a NN function f with n layers is equal to a NN with one layer. Thus, in NNs, to get benefits of increasing the number of layers, each layer has an activation function which makes the layer function f_n non-linear. In Figure 2.1, the network has three layers and the outputs of last layer (f_3) are the labels.

In a Convolutional Neural Network (CNN) [12, 32, 33], the network benefits from convolutional layers which are different from the fully connected layers in Figure 2.1. Figure 2.2 shows the structure of a typical CNN. The convolutional layers extract useful features that contain the spatial information of their input. In each CNN model, the convolutional layers are followed by at least one fully connected layer, shown with blue circles in Figure 2.2. CNNs are more effective when they become deep. In other words, increasing the number of layers will increase the number of parameters and improve the feature extraction power of the model. It is worth noting that the number of needed training samples is related to the number of parameters of the network.

Deep Convolutional Neural Networks (DCNNs) [12, 33, 34] are the state of the art in many areas of machine learning and computer vision. One of the first CNN models is the LeNet [26] architecture, with only three convolutional layers and proposed for digit recognition. LeNet is not a DCNN model and it fails to work well on bigger datasets. After LeNet, in 2012 AlexNet [35] was proposed and it was the best solution of the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) [35] at that time. AlexNen has 5 convolutional layers and three fully connected layers. In 2014 the VGG [13] network was proposed. VGG has 19 layers and it was the state of the art network architecture in 2014. Over time, the network architectures became deeper and deeper and training a network with millions of parameters was a big challenge due to some problems such as high computational complexity and vanishing gradients [36], limiting the network depth to about 22 layers. In 2016, ResNet [14] was proposed; the residual blocks [37] of ResNet architecture work like a shortcut path between the layers and makes it possible to train a network with more that 150 layers. Current networks benefit from the ResNet structure and many of them are a combination of different variants of ResNet and other architectures.

For training a DCNN a loss function must be defined. For an input x, a loss function takes the models output $f(x)$ and the label y and calculates the error of the model's prediction. To train a network, the model's prediction error on the training data (the loss function) should be minimized. A DCNN is trained using batch training such that the network does a loop over a batch of data $\{X', Y'\}$ and each time takes the derivative of the loss function $L(f(X'), Y')$, calculates the gradients, and back-propagates [38] them to the network to update its parameters. Currently, DCNNs surpassed human performance on

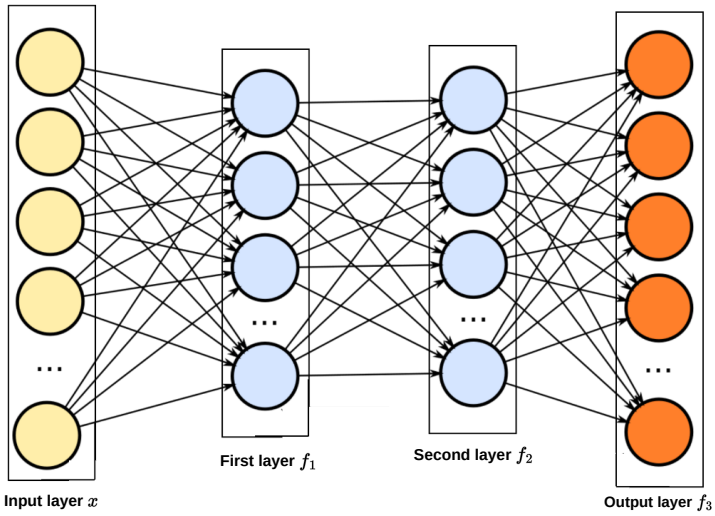

Figure 2.1: An example of a Neural Network structure, consisting of four fully connected layers.

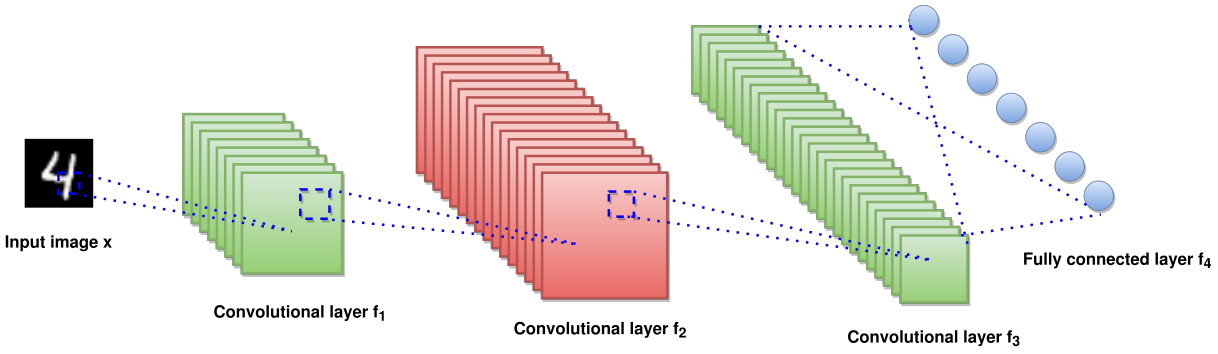

Figure 2.2: A Convolutional Neural Network structure. The convolutional layers act as feature extractors and extract the spatial information of their input. In the last stage, the network has a fully connected layer which is used for classification.

many supervised problems such as image classification [39] but remains a big gap between human performance and DCNNs in real-world scenarios, in particular in online learning [40, 41].

2.2 Online Learning

In supervised learning, a pair of training data (x_t, y_t) come from a joint distribution $p(x, y)$. The space of input and output are $\mathscr{X}$ and $\mathscr{Y}$ respectively. The learning procedure tries to find the optimal function f from the space of hypothesis functions $\mathcal{H}$. Thus, $f \in \mathcal{H}$ and f is a mapping function $f : \mathscr{X} \to \mathscr{Y}$ [40]. The learning procedure tries to minimize the following expected loss to find the optimal f:

$$E[L(f(x), y)] = \int L(f(x), y) dp(x, y) \tag{2.2}$$

where L is the loss function and can be the Euclidean distance between the output of f and the label y.

$$L(f(x), y) = ||y - f(x)||^2 \tag{2.3}$$

In the learning, the learning procedure has access to all of the training data (x, y) in advance.

DCNNs, as the state-of-the-art in machine learning, fail in online learning [40, 41] since they can not do training and testing at the same time with the same performance and efficiency of offline training. The main reason for that difference is the huge amount of time that a DCNN takes to be trained. In traditional offline learning, the whole training data are available and the model can be trained over multiple epochs (iterations on data). However, in online learning, the model f^t, which is the model f after training on avaiable data at time t, is learnt based on the current sample (x_t, y_t) and the preceding models f^{t-1}. Training a big model online using one sample is a highly challenging task. There are some solutions for this problem. First, the current sample (x_t, y_t) can be added to a set containing all of seen data and the model f^{t-1} can be trained or fine-tuned on that set. But the first solution requires an unlimited memory size. Thus, the other solution for online learning is to use a dynamic memory [42] containing a subset of previously visited data with an update mechanism instead of whole training data.

There is an assumption in some machine learning problems that the data used for training and evaluation are independent and identically distributed (i.i.d.), which makes

the problems easier to solve. The i.i.d. assumption is an important assumption of the central limit theorem, which indicates the probability distribution of i.i.d. samples with a finite variance reaches a Gaussian distribution. When the Gaussian assumption is correct, we can choose a linear model to solve the problem. However, the data in this thesis are not from a Gaussian distribution, and the solution models (CNNs) are also non-linear. Next, continual learning will be explained and discussed in details.

2.3 Continual Learning

Continual learning [4, 6, 43] is a learning method of a sequence of data comprising different tasks and domains. The i.i.d. assumption (independence and identically distributed) is divided into two parts. The first part is the independence assumption states each data point is drawn independently of the others, implying that the occurrence of other data points has no effect on the probability of each data point being drawn. In both online and continual learning, the independence assumption could be considered. The second part, however, assumes that the data distribution remains constant over time, which may not be true in continual learning because the data distribution can change over time for a variety of reasons, such as concept drift or domain shift [9]. An example of domain change in a continual learning problem is training a CNN model to classify different objects after the CNN model has already been trained on a class of "cats" images prior to a class of "flowers" images. Considering the absence of "cats" in the training process of the model on "flowers", there is a shift on the domain of training data. In the context of continual learning, the i.i.d. assumption is not true. In continual learning, the model should be able to adapt to new data from various distributions or domains. This necessitates that the model be capable of dealing with potential concept drift and adapting to the new data distribution without forgetting previously learned knowledge.

Assume, in the supervised learning scenario which is defined in Section 2.2, the training data consist of T different tasks (probability distributions) where $p = \{p_1, p_2, ..., p_t, ..., p_T\}$. Here, tasks could be considered to be different image classification classes ("cats" , "flowers"). As such, each time Equation 2.2 is minimised on the data (x, y) that come from the joint probability distribution $p_t(x, y)$ of the task t. An ideal continual learning method should possess the following three properties:

1. It models current tasks data p_t, building on the knowledge of previous tasks $(p_1, p_2, ..., p_{t-1})$.
2. It has the ability to improve or maintain performance on the previously learned tasks while learning a new task.

3. It is scalable, handling small and large numbers of tasks in different scenarios.

Based on these properties, it is possible to categorize the continual learning methods from different perspectives into two streams:

- The first stream is based on Bayesian frameworks [44, 45, 46] and including prior-focused, likelihood-focused, and hybrid approaches. This stream formulates the continual learning problems from a Bayesian perspective. The goal is to maximise the posterior probability of model's parameters given data. However, the posterior evaluation is intractable and the posterior is calculated using the normalized multiplication of likelihood and prior. In prior-focused methods, the model's posterior of the previous task is used as a prior for the current task and also for likelihood-focused methods, the model itself is adapted by modifying the likelihood of the model. The hybrid approaches try to approximate the posterior of new task by using both prior and likelihood.

- A second stream [47, 48] categorises based on regularization, rehearsal and ensemble methods. In this stream, regularization methods try to add a constraint to the training phase of the model to mitigate the model's forgetting problem of previous tasks. On the other hand, the rehearsal methods keep some data samples of previous tasks and use them in the training of new tasks. Finally the ensemble methods point to the methods that expand the network or use a combinations of the mentioned methods (regularization and rehearsal) to solve the problem.

It is worth noting that the prior-focused methods are quite similar to regularization-based solutions. Additionally, the likelihood-focused and the hybrid methods are the same as rehearsal and ensemble methods. Thus, the main idea is the same in both streams, but each stream has a different perspective. This thesis is built on the first first stream which is the Bayesian framework. Prior-focused methods, such as Elastic Weight Consolidation (EWC) [5] and Memory Aware Synopses (MAS) [49], assume a previously-learned model's weights to be a prior for training the current network weights, responsible for learning a new task, and apply regularization during the training to prevent previous related weights from changing too significantly. In particular, these two strategies try to keep important parameters, associated with the previous tasks, fixed via a penalty term in the loss function. In general, in regularization approaches, the network regularized loss function $L_R(\Theta, f(x), y)$ is formulated as below, where a regularization term is added to the loss

function:

$$L_R(\Theta, f(x), y) = L(f(x), y)(2.3) + \lambda \sum_{k=1}^{K} \omega_k (\theta_k - \theta_k^*)^2, \tag{2.4}$$

here R in L_R stands for regularized loss function, Θ is the set of all model's parameters where $\Theta = \{\theta_1, \theta_2, ..., \theta_k, ..., \theta_K\}$, L (2.3) is the loss for training the current task, λ is the regularization coefficient, ω_k is the *parameter importance* for the k-th parameter across all previous tasks, θ_k^* is the k-th weight learned from the previously seen tasks, and K is the number of weights (parameters) in the model. ω_k quantifies the importance of the parameters and it is calculated differently in various algorithms. For example, parameter freezing [50] can be considered as one extreme in regularization approach. In EWC [5], ω_k is calculated as the diagonal of the Fisher information matrix, while in MAS [49] ω_k is calculated as the gradient of l_2-norm of the neural network output given the inputs. The calculated gradients in the back-propagation step of training the model are used in both EWC and MAS for determining important parameters of the training model. Sparsity constraint methods [51, 52, 53] are other examples of regularization approach, whereby an l_1-norm constraint on the network weights during training preserves certain weights. In other words, the training procedure should not be selfish, and specifies a part of the model for training future tasks. Sparsity allows the model to leave enough room for other tasks to be learned.

The likelihood-focused and rehearsal methods perform similarly, wherein the methods focus on maximizing the likelihood function by incorporating previous data information. Examples such as deep generative replay (DGR) [54] and variational generative replay (VGR) [46] involve storing past data or training generative models for past tasks and then using stored information to train new tasks. For instance, Shin *et al.* [54] used generative adversarial networks (GANs) to generate data from each task as examples to be utilized during the training of new tasks.

Likelihood-focused methods are interpreted as rehearsal, since the algorithm is able to *rehearse* with examples from the previous tasks during the training phase of new task via stored information. Eventually this way, it maximizes the likelihood function across all of the tasks, as described in [45]. These methods are the most effective solutions, given a large data set and in the absence of any memory constraint. Despite their superior performance, these methods are not scalable as memory requirement grows with the number of tasks.

Hybrid methods, as their name suggests, take advantage of a prior-focused technique and a likelihood-focused method at the same time. For example, variational continual learning (VCL) [45] combines the posterior from the previous task (i.e., the prior in the

current task) with information from the new task (i.e., its likelihood) through multiplication and normalization.

Progressive networks [55] and dynamic expandable networks (DEN) [56] increase the number of parameters in the neural network as the network learns new tasks. They combine these new parameters with those from the original model. In particular, in *progressive networks* the parameters were "frozen" to ensure that information related to the previous tasks was not lost, and then a new neural network of the same size was initialized with lateral connections to the previous network. A limitation to this approach is that the number of parameters increases linearly with the number of tasks. DEN tries to address this issue by dynamically varying the number of parameters, addressing the issue of continual growth in the number of network parameters by applying sparsity regularization and removing those weight which not activated frequently.

In continual learning problems the tasks may have discrete labels such as class names or categories and the model can benefit from the task's labels in the training or even testing stage. In the literature, the majority of solutions address the continual learning problems with divided tasks, where the task boundaries are known. However, for a more realistic and real-world scenario the online-continual learning is defined.

2.3.1 Online-Continual Learning

An online-continual learning [25, 27] is an integration of online learning and continuous learning. Typically, continual learning scenarios are classification-based scenarios in which each task is defined by a class and contains a fixed number of images of a specific subject and domain; however, in online learning scenarios, there is usually a stream of data that becomes accessible to the model at each time step. Parisi et al. [27] described some desiderata for an online-continual learning problem:

- Streaming data: It is assumed the data are never-ending stream and temporaly correlated.
- Unsupervised in terms of task labels: The task boundaries and labels are not provided for training.
- Limited resources: The number of tasks on the data stream can be infinite; however, if the computational complexity and the required memory be directly proportional to the number of tasks, the solution would be unbounded.

- The training data stream are non-i.i.d.

The solutions for online-continual learning are usually rehearsal methods. iCARL [57] as a rehearsal method selects some samples and features from each task and puts them in a replay buffer to use them for future training. The method calculates the average of the selected features and uses them in the nearest neighbor algorithm on the feature space of the network to do the classification. Additionally, the iCARL method uses knowledge distillation [58] to keep the model's parameters unchanged. To do so, before training a new task, the model stores the logits of the selected replay samples of previous tasks and tries to keep them unchanged during training using a distillation loss. Another rehearsal method, GEM [59], uses the last samples of each task and put them in replay memory and uses them to modify the gradients of the current task to mitigate the catastrophic forgetting. But these two well-known methods do not update their reply-buffer and they would have problem of exploding memory when the number of tasks increased. Currently two methods [60, 61] are supposed to update their replay buffer to keep the distribution of the replay-buffer the same as the distribution of the data which is currently seen by the model.

In [25] a gradient-based sample selection method is proposed for online-continual learning. Based on the constrained optimization view of continual learning, they formulate sample selection as a constraint reduction issue. The objective is to choose a fixed subset of constraints that best approximate the feasible region outlined by the initial constraints. They also demonstrate that choosing a fixed subset of constraints is equal to maximizing the diversity of samples in the replay buffer with the feature parameters gradient. They evaluate their method on MNIST [26] and CIFAR10 [62] datasets.

Continual learning methods are typically tested on classification datasets, like MNIST [26], CIFAR10 [62], and ImageNet [15]. Another dataset that specifically designed for continual learning is Core50 [63] which has been used to assess the performance of continual learning algorithms because it contains a diverse collection of classes and images that can be used to mimic a stream of data.

All the mentioned datasets for cintinual learning scenarios are classification datasets. A classification dataset usually is fed to the model as a sequential stream of data in online-continual learning evaluation scenarios [25]. Thus, the datasets and their related testing scenarios share the issue of having few practical real-world applications; however, video data has the potential to be a suitable data for the problem of continual learning. This study chooses Video Object Segmentation (VOS) and its related datasets from various video analysis methods to assess the effect of proposed continual learning solutions. VOS issues and solutions are addressed in the following section.

2.4 Video Object Segmentation (VOS)

Video object segmentation (VOS) aims to extract an accurate pixel-wise object mask in each frame of a given video. VOS has many real-world applications, such as video summarization, human computer interaction, and autonomous vehicles [64]. Broadly, proposed VOS algorithms can be divided into two different streams: i) semi-supervised or one-shot VOS, when the ground truth masks of the target objects are provided in at least one frame at inference time, and ii) unsupervised VOS, when no information about the objects is provided. The focus of this thesis is on the former context, that of semi-supervised VOS.

The intuition behind semi-supervised VOS is to perform fine-tuned learning on a VOS model, separately for each test video, based on the given target information (i.e., the given object mask). However, training a big model on a given frame and mask of each testing video is a challenging task since it requires fine-tuning of a big model on very frame of a video which is computationally expensive and not efficient. Early solutions in the literature [65, 66] fine-tuned a pretrained VOS on the given information in a video at evaluation time. This idea is not feasible, due to the limited training samples, the VOS model size, and the time-consuming training process. In practice, online learning-based VOS approaches [16, 17, 20, 22] address these challenges by introducing efficient training mechanisms, specifying a part of model for online updating, and keeping some amount of information in memory to augment the training set for model fine-tuning.

Approaches to VOS can be split up into three distinct streams, including Online VOS, matching-based VOS, and propagation-based VOS. Having a memory to keep and use information (features and segmented masks) about the evaluated frames is a feature that is shared by the most advanced methods in each of these three streams, which together form the memory-based VOS broad category. In addition to the memory-based VOS and its three streams, the VOS solutions can be divided into two categories: those that are designed for and function most effectively with short videos, and those that are designed for and work best with long video datasets. Following this categorization of VOS methods, a discussion of memory-based VOS solutions will take place.

2.4.1 Memory-based VOS

Memory-based approaches [16, 17, 20, 21, 22, 67] try to address semi-supervised VOS problems by storing feature representations and predicted output masks of preceding frames in a memory and use them for evaluating the current frame.

Using this strategy, there are different approaches proposed to retrieve information from this dynamic model's memory.

A first solution is to propagate the information of the most recent frames received from the predicted masks [66] or a hidden representation proposed by the recurrent methods [68, 69].

A second solution is to update (fine-tune) a small model on the memory proposed by the online learning methods [16, 22, 23, 65, 70].

A third solution is to match the representations of previous frames stored in the memory with the corresponding features extracted from the current frame proposed by the matching-based methods [20, 71, 72, 73, 74, 75, 76, 77, 78].

Each stream will be discussed in detail. The propagation-based VOS solutions are discussed next.

Propagation-based VOS

In propagation-based VOS techniques [79, 80, 81], the segmented masks of previous evaluated frames are repeatedly propagated to the current frame using propagation-based VOS techniques [79, 80, 81].

The optical flow VOS methods align the object segmentation mask from the previous frame to the current frame using the estimated optical flow vectors [82]. Following that, the aligned segmentation mask is utilized to initialize the current frame, and the mask is refined further using additional data such as appearance features. Following studies focus on offline learning by employing optical flow to convey temporal information [80, 83, 84]. RVOS [69] employs recurrent neural networks (RNNs) to analyze both spatial and temporal knowledge. The spatial recurrence allows the model to identify and distinguish between different object instances within a single frame. Meanwhile, temporal recurrence allows the model to maintain consistency in the segmentation of these objects across multiple frames over time.

Despite the encouraging findings of propagation-based VOS methods, these techniques are frequently susceptible to error buildup caused by occlusion or drifting.

Matching-based VOS

Matching-based methods try to send a query contains the encoded information of current frame to the memory and extract some useful information helping segmenting the current

frame. Among the matching-based methods is the STM [20], which uses a similarity matching algorithm to retrieve encoded information from the memory and pass it through a decoder to produce an output.

In VOS the target object in the query frame usually appears in the local neighborhood of the target's appearance in the memory frames, but STM is based on non-local matching between the query and memory frames. Therefore, to solve this problem, KMN [77] proposed a kernelized memory network applying a Gaussian kernel to address the non-localization aspect of the STM.

HMMN [78] also proposed kernel-based memory matching to achieve temporal smoothness by restricting possible correspondences between two adjacent frames to a local window and applying a kernel guidance to the non-local memory matching. For matching of distant frames, HMMN applies tracking of the most probable correspondence of a memory pixel to a query pixel. Instead of building a specific memory bank and therefore affinity for every object in the video as in STM, STCN [73] builds a model, which learns all object relations beyond just the labeled ones by using an affinity matrix based on RGB relations. For querying, a target object passes through the same affinity matrix for feature transfer. To deal with appearance changes and deformation, LCM [71] proposed applying a memory mechanism to retrieve pixels globally, and to learn position consistency for more reliable segmentation.

The current state-of-the-art methods [21, 85] for VOS follow the STM structure and are matching-based methods.

Online VOS

Another stream of memory-based VOS methods is online learning-based VOS which learns the new object appearance within an online learning-based approach [16, 17, 28] simultaneously at inference time. In this category of solutions, instead of using a matching-based (matching based) algorithm on each frame, a small latent model network, the so-called target model, is updated every s frames which is eventually used to learn the updated information stored in the memory for segmenting the proceeding video frames.

The target model proposed by FRTM [22], LWL [16] and the induction branch of JOINT [17] is formulated as a small convolutional neural network, which performs online learning on the available training data in the memory $\mathcal{M}$. As such, these methods can provide an efficient yet effective dynamic update process for VOS frameworks. The target model structure for LWL and JOINT is a two layers CNN with the kernel size of 1×1 for

the first layer to do dimensionality reduction and a kernel size of 3×3 for the second layer for training on the memory.

While target model-based approaches improve the performance of VOS, the effectiveness of online learning algorithms is highly dependent on their memory capacity and usage. In other words, to obtain the best performance, these models require to store all preceding output masks and the encoded features in their memory and also make a way to increase the generalization of the updated model. Therefore, memory limitation results in facing similar challenges already known in the domain of continual learning and was discussed in Section 2.3.1. This thesis hypothesizes that these issues can be mitigated, which is motivated by the success of continual learning algorithms in preserving learned knowledge while limiting the required memory. The approach proposed in this thesis stems from Online VOS.

2.4.2 Video Sequence Length

Recent proposed VOS solutions are mainly designed for short video datasets. Three well-known VOS datasets are DAVIS16 [86], DAVIS17 [87], and the YouTube-VOS18 [88] datasets. The DAVIS16 [86] validation set has 20 videos, each of which has a single object for segmentation; the validation set of DAVIS17 [87] contains 30 video sequences with multiple objects to be segmented in each frame. The validation set of YouTube-VOS18 has 474 video sequences of 65 seen (which are present in the training set) and 26 unseen object classes. Figure 2.3 shows three video sequences from the DAVIS2016 dataset, where we can see that target objects do not have an abrupt change through video frames. Objects could have small changes, such as in the "cow" video (the longest video in DAVIS2016 at 104 frames), and the other two videos (soapbox and motocross-jump) possess variations in object appearance, however the changes are gradual. As a result, for such datasets the identically distributed assumption of frames is usually valid, particularly for short videos. It is thus worth mentioning that the YouTube-VOS18 sequences are even shorter than those in DAVIS16 and DAVIS17, where the longest video in the validation set of YouTube-VOS18 has 36 frames. Considering these short video sequences for evaluating a VOS solution makes a specific design for each VOS solution. For example, as is clearly mentioned in [21], STM [20] is not computationally efficient to handle long video sequences. Another example is JOINT [17], where its transformer branch does not work efficiently on large video sequences [21].

The semi-supervised VOS approaches maintain the identically distributed assumption, despite the fact that i.i.d. assumption is clearly not valid in all video sequences, particularly

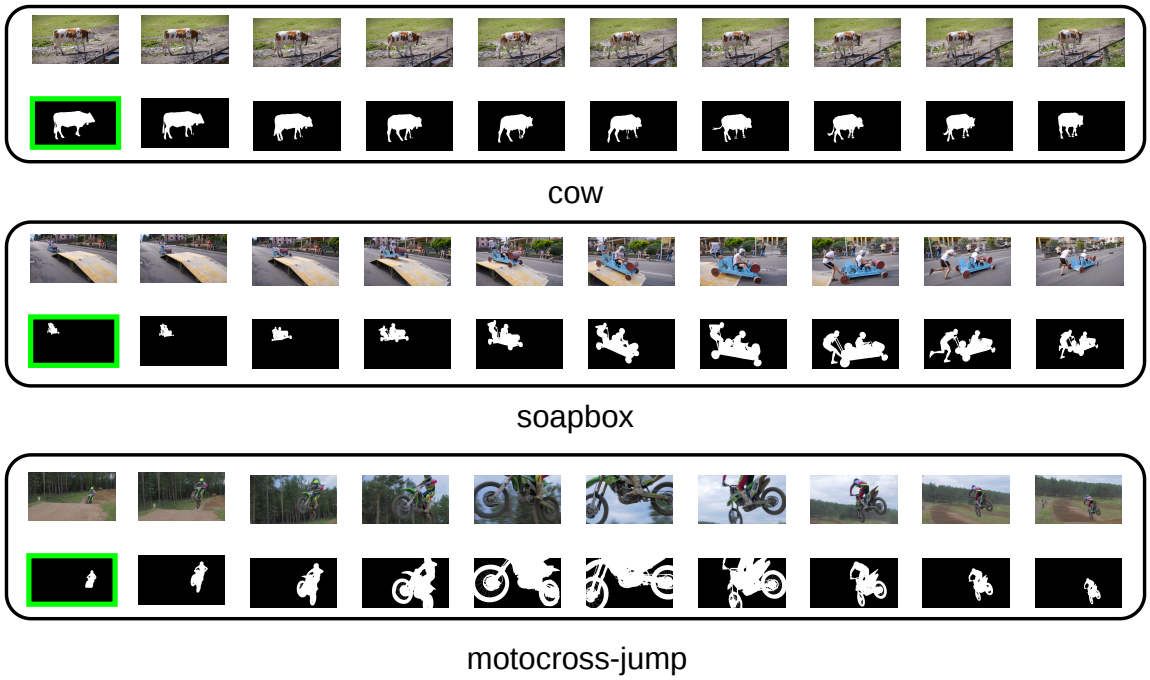

Figure 2.3: A set of sub-sampled frames from three videos of the DAVIS16 dataset [86], in each case two rows: actual images (top) and segmented objects (bottom). The first video, "cow" is the longest in DAVIS16, however there is no significant change between frames. There is a gradual change in appearance in the other two videos. The given annotated (ground-truth) frame in each video is highlighted in green.

longer ones. It is precisely for this reason that state-of-the-art semi-supervised VOS models are not expected to have similar performance on long video datasets [21] in comparison to their performance on short video datasets.

Figure 2.4 shows the "dressage" video from the Long Videos dataset [1], which consisting of three long sequences with a total of 7411 frames. The dataset has 21 labeled frames for each video for evaluation. As is clear from Figure 2.4, an i.i.d. assumption is not at all valid on the "dressage" video, because of the 22 substantial distribution drifts which take place, a behaviour which is much more closely aligned with the *non*-i.i.d. assumption of continual learning. The labelled masks in the Long Videos dataset [1] does not cover all the challenges of the dataset as shown in Figure 2.4. It is worth noting that the evaluation label mask is chosen uniformly in the Long Videos dataset.

Long video sequences containing several concepts are more challenging to be learned since the memory-based VOS model requires a memory with large capacity to store the previously learned frames representations.

To address the limitations in memory and training time of an online solution, AFB-URR [1] uses an exponential moving averages to merge a new memory component with earlier ones if they are similar, or to store it as a new component in the memory otherwise. The model removes unused features from the memory when its capacity reaches a predefined limit.

Using a global context module [89] is another way to deal with the limitations caused by long video sequences. The model calculates a mean of the entire memory components and apply it as a single representation.

However, both methods apply a compact representation of the memory, which sacrifices the segmentation accuracy [1, 21, 89]. On the other hand, XMem [21] uses a multi-store feature memory to avoid compression and achieves much higher accuracy in both short-term and long-term predictions.

The current state-of-the-art VOS method on both long and short video datasets is ISVOS [85]. They propose a new method for Video Object Segmentation (VOS) that combines instance understanding and matching-based VOS. They recommend a two-branch network, with one branch focusing on instance segmentation (IS) and the other on VOS. The network uses object queries in the IS branch to obtain instance-specific information about the current frame. This data is then included in the query key used by the VOS branch, which performs spatial-temporal matching with a memory bank. As a result, an instance-augmented matching approach is developed that improves VOS accuracy. To produce the final segmentation results, they include a multi-path fusion block that combines memory readout with multi-scale features from the instance segmentation decoder.

In this thesis, we focus on improving Online VOS by providing an efficient memory usage method (RMSCL) and a regularization based continual learning approach (GRCL).

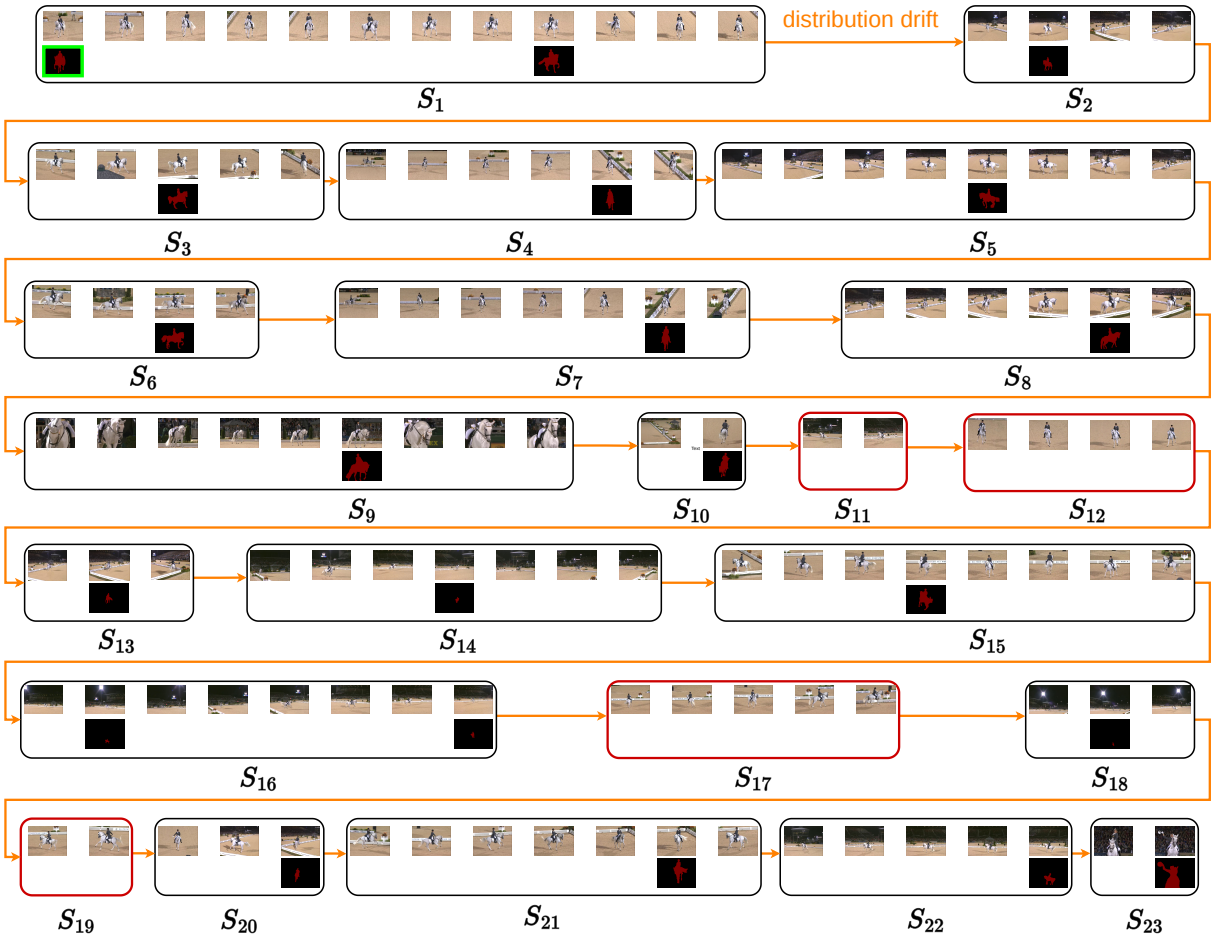

Figure 2.4: A subset of frames from "dressage" video of the Long Videos dataset [1]. The video consists of 23 sub-chunks that are separated from each other by significant distributional drifts or discontinuities. The lower (sparse) row, in each set, shows the annotated frames. The four sub-chunks that do not have any labelled frame in the evaluation set of Long Videos dataset are encircled in red.

Chapter 3

Problem Formulation

Current online Video Object Segmentation (VOS) methods [16, 17, 20, 22] aim to segment target objects from background in every frame of a video sequence and they suffer from many limitations, usually on long video sequences. In this chapter, first, the general Online VOS framework is formulated and explained in Section 3.1. The main online learning part of this framework, the "target model" C, as is suggested in [22], is specifically discussed. This framework is adopted in all of the contributions to this thesis. Finally, limitations and the related motivations of proposing the continual learning-based VOS solutions are explained in Section 3.2. The proposed problem formulation is used in the next few chapters.

3.1 General Online VOS Model

An Online VOS model [16, 17, 22] usually has a U-Net structure [90] as is shown in Figure 3.1. The goal of Online VOS is to segment an object from each image frame F of a video and provide a segmented mask Y which is a binary segmented image such that the pixels belonging to the object are labelled 1, and the background pixels are labelled 0. Typically, a general Online VOS model comprises the following pieces:

1. A pretrained encoder, extracting feature X from each frame F;
2. A memory $\mathcal{M} = \{\mathcal{X}, \mathcal{Y}\}$, storing a set of features $\mathcal{X} = \{X_0, X_1, ..., X_{t-1}\}$ and their associated labels $\mathcal{Y} = \{Y_0, Y_1, ..., Y_{t-1}\}$ masks that could be updated with input feature X_t and estimated output Y_t at time t;

3. A target model C^t, which potentially is trained on the memory $\mathcal{M}^t$ at every time t, and provides information to the decoder D;

4. Pretrained decoder D and the label encoder E [16], networks which obtain temporal information from the target model alongside the encoder's output, to generate a fine-grain output mask Y from its associated frame F.

Here, the time index t is based on input time frame. Thus, at time t, C^{t-1} is updated to C^t on $\mathcal{M}^t$. Next, the output Y_{t+1} is estimated using C^t and then $\mathcal{M}^t$ can be updated with pairs of (X_{t+1}, Y_{t+1}) to create $\mathcal{M}^{t+1}$. Potentially, we could update $\mathcal{M}$ every time frame t, but there is also a possibility to update the memory every $\Delta_{\mathcal{M}}$ frames. Considering t indicates time frame, if $\Delta_{\mathcal{M}} > 1$ then there are some time steps that the memory $\mathcal{M}$ remains constant. The same thing could happen on the target model C when $\Delta_{\mathrm{C}} > 1$. These concepts are defined in the lines 5 and 12 of Algorithm 1. In addition to $\Delta_{\mathcal{M}}$ and Δ_{C}, two update time indices $t_{\mathcal{M}}$ and t_{C} are defined as the most recent update time index of memory $\mathcal{M}$ and target model C, respectively. This process is explained in Algorithm 1. There is an order for updating target model C and memory $\mathcal{M}$, which is explained in Algorithm 1 and is depicted in Figure 3.1. Thus, for segmenting the current frame F_{t+1}, the memory $\mathcal{M}^t$ is the most updated version of itself; it should be checked if the target model needs to be updated, and then the segmentation could be done using the most updated target model C^t.

An Online VOS model S_Ξ is first trained offline to minimize the following loss function and find the model parameters Ξ:

$$\Xi = \arg\min_{\Xi'} \mathcal{L}(S_{\Xi'}(F), Y). \tag{3.1}$$

In Equation (3.1), $\mathcal{L}$ is usually a pixel-wise cross entropy loss [91], F is an image frame and Y is the segmented mask. All of the parameters of the Online VOS model (Ξ) are trained offline; however, some parameters of the model Θ are supposed to be updated online at the evaluation time.

For the general Online VOS model described earlier in Section 3.1, the parameters of the target model C are called Θ, which are primarily the target model's convolutional filter weights, and $\Theta = \{\theta_l\}_{l=1}^{K}$, where K is the total number of parameters of the target model C. It is worth noting that Θ is a small subset of Ξ, the whole framework's parameters. For example in a framework including an encoder, a decoder, and a target model, Θ can be the set of parameters of the target model, while Ξ is a set including all of parameters of the encoder, decoder and the target model. Θ is trained online on the memory, and the target

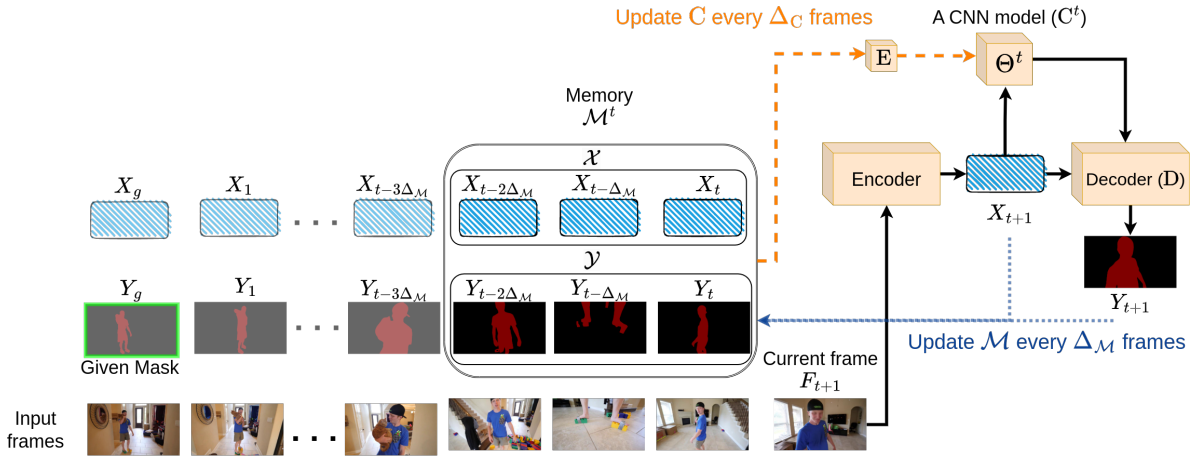

Figure 3.1: General Online VOS framework: The target model C^{t-1} is updated on memory $\mathcal{M}^t$ to form C^t. The dashed lines shows how the target model C is updated based on memory $\mathcal{M}$ every Δ_C frames. Memory $\mathcal{M}^t$ is updated every Δ_M frames with new information (X_{t+1}, Y_{t+1}). The dotted lines show the memory update. The proposed methods in this thesis are mainly engaged with the target model component (C) of the framework. The frame images used in the figure are taken from the Long Videos dataset [1].

model C is usually a small convolutional neural network, for reasons of efficiency. The target model is updated every Δ_C frames throughout the video, repeatedly trained on feature $\mathcal{X}$ and their associated encoded labels $E(\mathcal{Y})$ based on stored decoder outputs $\mathcal{Y}$ from preceding frames. Both $\mathcal{X}$ and $\mathcal{Y}$ are stored in memory $\mathcal{M}$, where the memory is constrained to some size N. In Figure 3.1, the maximum memory size is limited to 3. It is worth noting that E is a label encoder, generating sub-mask labels from each Y [16]. For online training of C^t at time t, every $Y \in \mathcal{M}^t$ is fed to E and $E(Y)$ generates some encoded masks that the target model C^t tries to learn. In other words, the target model C^t learns what E specifies from each Y given X as the input. Thus, given input feature X_t the output of trained target model C^t is an estimation of $E(Y_t)$. The target model acts like a dynamic attention model to generate a set of score maps $E\big(C^t(X)\big)$ in order for the segmentation network (D) to produce the segmented output mask Y associated with each frame F. The loss function L which is used for the online training of target model at time

Algorithm 1: General Online VOS.

Input: $\{X_{t+1}, \mathcal{M}^t, \Theta^{t-1}, \mathrm{D}, \mathrm{E}, \Delta_\mathrm{C}, \Delta_\mathcal{M}, t_\mathrm{C}, t_\mathcal{M}\}$

// Updating the target model C^{t-1}.

if $t - t_\mathrm{C} = \Delta_\mathrm{C}$ **then**

  $t_\mathrm{C} \leftarrow t$

  // Updating the target model C^{t-1} parameters using Equation (3.2).

  $\Theta^t \leftarrow L(\Theta^{t-1}, \mathcal{M}^t)$

else

  $\Theta^t \leftarrow \Theta^{t-1}$

end

// Segmenting the object using the decoder D.

$Y_{t+1} \leftarrow \mathrm{D}\Big(X_{t+1}, \mathrm{E}\big(\mathrm{C}^t(X_{t+1})\big)\Big)$

// Updating the memory $\mathcal{M}^t$.

if $t - t_\mathcal{M} = \Delta_\mathcal{M}$ **then**

  $t_\mathcal{M} \leftarrow t$

  $\mathcal{M}^{t+1} \leftarrow \mathcal{M}^t \cup \{X_{t+1}, Y_{t+1}\}$

else

  $\mathcal{M}^{t+1} \leftarrow \mathcal{M}^t$

end

return $\{Y_{t+1}, \mathcal{M}^{t+1}, \Theta^t, t_\mathrm{C}, t_\mathcal{M}\}$

t is

$$L(\Theta^t, \mathcal{M}^t) = \sum_{n=1}^{\left|\mathcal{M}^t\right|} \Big\| d_n W_n \Big(\mathrm{E}(Y_n) - \mathrm{E}\big(\mathrm{C}^t(X_n)\big) \Big) \Big\|_2^2 + \sum_{k=1}^{K} \lambda \, {\theta_k^t}^2, \tag{3.2}$$

where $\left|\mathcal{M}^t\right|$ is the number of feature and mask pairs $\{X, Y\}$ in the memory $\mathcal{M}^t$.

Depending on the overall architecture, E is an offline / pre-trained label encoder network, as in [16], or just a pass-through identity function, as in [22]. It is worth noting that the influence and effect of E is not the focus or interest of this thesis.

In Equation (3.2), W_n is the spatial pixel weight, deduced from Y_n, and d_n is the associated temporal weight decay coefficient. In the loss function $L(\Theta^t, \mathcal{M}^t)$, W_n balances the importance of the target and the background pixels in each frame, whereas d_n defines the temporal importance of a pair of feature and mask (X_n, Y_n) in memory, typically emphasizing more recent frames [16].

In another point of view, each frame F is fed to the Online VOS framework, and a pair of extracted feature X and the estimated segmented object mask Y become available for updating the framework. In Online VOS frameworks, it is usually assumed that F is identically distributed, which is one of the i.i.d. properties; however, in general, a new input frame F can be **non**-identically distributed from some jointly distributions of video frames that form a continual learning problem [6].

In most video sequences, the given input frame (F) can have a sudden change from previous frames. Thus, we need to get benefit from all the available data while doing evaluation. The most efficient model for updating the parameters Θ^{t-1} of an Online VOS framework given all available (seen) data $\mathbb{D}^t$ at time t is formulated as Maximum A Posteriori (MAP) [92] of $p(\Theta|\mathbb{D}^t)$. Since finding the true posterior probability is not tractable, the following optimization based on Bayesian theory is suggested

$$\Theta^t = \arg\max_{\Theta} p(\Theta|\mathbb{D}^t) = \arg\max_{\Theta} \left(\frac{p(\mathbb{D}^t|\Theta)p(\Theta)}{p(\mathbb{D}^t)} \right), \tag{3.3}$$

where $p(\mathbb{D}^t|\Theta)$ is the likelihood and $p(\Theta)$ is the prior. This Bayesian interpretation of the target model's updating step C serves as a framework for proposing the contributions of this thesis. In the next section, the limitations of Online VOS and the proposed methods are addressed using the Bayesian interpretation of Online VOS.

3.2 Limitation and Motivations

Online VOS methods suffer from three main limitations [16, 21] which deteriorate their performance, particularly on long videos:

1. **Memory Size:** To maximize performance, Online VOS would need to store all or most of the extracted information of preceding frames $\mathbb{D}^t$ in the memory $\mathcal{M}^t$. However, for videos of arbitrary length this requires an unlimited memory size, which is infeasible.

2. **Target Model Updating:** Even with an unlimited memory size, updating the target model C on an arbitrarily large memory with many redundancies would be computationally problematic for an online solution.

3. **Robustness:** The sensitivity of Online VOS approaches to the selection of different hyper-parameters, such as the target model's step size Δ_{C} and memory updating step size $\Delta_{\mathcal{M}}$, which could mitigate both speed and accuracy.

The proposed contributions of this thesis address these limitations by incorporating effective methods applied to the target model C and memory $\mathcal{M}$. Since video frame information is provided consecutively into the Online VOS framework and the i.i.d. assumption is not valid in video sequences, there is a high possibility of drift in the object's appearance, especially in long-video sequences. As such, the conventional approach of passing all of the information, as a whole, to the model to decide which to use, is not effective and can lead to ineffective learning or even divergence in the training process of the target model. Thus, all available data information $\mathbb{D}$ cannot be placed in the memory and we need to limit the memory size.

Given the first two mentioned limitations (memory size and the target model updating), we need to be able to handle long video sequences; the solution to do this is to limit the memory size. Taking this limitation into account, when the memory reaches its maximum capacity, we must remove some data from the memory to make room for new incoming data. In this case, all available data information could be regarded as the sum of all previous memories $\mathbb{D}^t = \bigcup_{l=1}^{t} \mathcal{M}^l$, and the most recently updated memory version would be a subset of all the visited information $\mathcal{M}^t \subset \mathbb{D}^t$. Considering the memory size limitation, Equation (3.3) is re-written as

$$\Theta^t = \arg\max_{\Theta} p(\Theta|\mathbb{D}^t) = \arg\max_{\Theta} \left(\frac{p(\bigcup_{l=1}^{t} \mathcal{M}^l|\Theta)\ p(\Theta)}{p(\mathbb{D}^t)} \right). \tag{3.4}$$

Maximizing Equation (3.4) over Θ does not have anything to do with $p(\mathbb{D}^t)$, thus we can remove the denominator. Additionally, the available information is limited to the last updated memory $\mathcal{M}^t$ assuming $\mathbb{D}^t$ is equal to $\mathcal{M}^t$.

Thus, the Bayesian formulation for updating target model C would be

$$\Theta^t = \arg\max_{\Theta} \log p(\Theta|\mathbb{D}^t) \simeq \arg\max_{\Theta} \Big(\log p\big(\mathcal{M}^t|\Theta\big) + \log p(\Theta) \Big). \tag{3.5}$$

Optimizing Equation (3.5) equals minimizing loss function in Equation (3.2), thus

$$\Theta^t = \arg\max_{\Theta} \log p(\Theta|\mathbb{D}^t) \simeq \arg\max_{\Theta} -L(\Theta, \mathcal{M}^t). \tag{3.6}$$

Given Equation (3.5) and inspired by continual learning [4, 6, 43], there are three different solutions which are proposed in this thesis:

1. **Prior-focused:** To maximize the posterior probability, we can focus only on the prior $p(\Theta)$ and simply limit the memory size N and remove the old data from the memory.

To this end, the posterior $p(\Theta|\mathbb{D}^{t-1})$ of previous update could be considered as prior $p(\Theta)$ for current update. Inspired by continual learning [4, 6, 43], the knowledge learned during preceding updates could be helpful to overcome the memory limitation problem. To achieve this goal, the parameters, Θ^{t-1}, of the target model C^{t-1} are regularized in each online learning step, with a goal of preserving the prior knowledge, acquired from those earlier information that might not be presented in the memory $\mathcal{M}^t$. The proposed prior-focused solution will be defined in Chapter 4.

2. **Likelihood-focused:** The other solution for this problem is to focus on the likelihood probability $p\big(\mathcal{M}^t|\Theta\big)$ by providing the best possible $\mathcal{M}^t \subset \mathbb{D}^t$ using a memory and a memory selection method or a data generation model. The provided likelihood-focused method helps to improve the efficiency of the solution inspired by rehearsal methods [46, 54] in continual leaning. The goal of the likelihood-focused solution is to target the second limitation of Online VOS, which is the target model updating for a small number of epochs on a large memory, as explained in Chapter 5.

3. **Posterior-focused:** The final proposed solution benefits from both likelihood- and prior-focused solutions and tries to find a hybrid method covering the limitations of the preceding solutions (prior-focused and likelihood-focused) to improve the robustness of Online VOS. The proposed posterior-focused solution is similar to hybrid methods [56] in continual learning and is discussed in Chapter 6.

Chapter 4

Prior-focused VOS solution

As discussed in Section 3.2, prior-focused approaches in continual learning rely on prior knowledge learned from previous tasks to learn a new task. In Equation (3.5), $p(\Theta)$ is the prior probability of target model's parameters Θ. A prior-focused approach tries to maximize the posterior probability, i.e optimizing Equation (3.5) by focusing on the prior $p(\Theta)$ to update Θ. To do this, we first need to understand the purpose of $p(\Theta)$ for Online VOS. Generally, the available prior knowledge for an Online VOS framework that is about to update its target model C^{t-1} at time t, is the collection of features from the seen frames $\mathcal{X} = \{X_l\}_{l=1}^{t}$ and their associated predicted object masks $\mathcal{Y} = \{Y_l\}_{l=1}^{t}$, which are placed in $\mathbb{D}^t = \{X_l, Y_l\}_{l=1}^{t}$ and $\mathbb{D}^t$ is the set of all visited data up to time t. However, in practice, a subset of this data is available in the memory $\mathcal{M}^t$ at time t, where $\mathcal{M}^t \subset \mathbb{D}^t$ due to memory size restrictions. In addition to the preserved information in the memory $\mathcal{M}^t$, the target model parameters Θ^{t-1} also implicitly carry information from previous updates of the target model. Thus, the prior model $p(\Theta^{t-1})$ is defined as $p(\Theta^{t-1}|\mathbb{D}^{t-1})$. Given available data $\mathcal{M}^t$ at time t, maximizing the Bayesian formulation of posterior probability in Equation (3.5) can be done by focusing on the prior $p(\Theta^{t-1})$ to regularize the important parameters in Θ^{t-1} while updating Θ^{t-1} to learn new information in $\mathcal{M}^t$. Those important parameters could be selected based on their calculated gradients in the preceding update steps, and will be explained in Section 4.1.

4.1 Gradient-based parameter regularization

Parameter regularization seeks to preserve important parameters of the target model, C, specifically those parameters which were learned or significantly modified (and therefore

important) in the preceding update steps. This parameter regularization will be added to the loss function of Equation (3.2) and will be minimized alongside the loss function.

In continual learning, Elastic Weight Consolidation (EWC) [5] and Memory Aware Synopses (MAS) [49] are two gradient based regularization methods. The MAS algorithm is formulated such that at update step t, the importance of each parameter θ_k^t is associated with its gradient magnitudes $\{u_k^l\}_{l=1}^{t-1}$ found during the back-propagation [93] in all preceding update steps. Therefore, during each online learning step, the parameter's gradient weight ω_k^t is updated based on the gradient magnitudes,

$$\omega_k^t = \omega_k^{t-1} + u_k^t \tag{4.1}$$

ω_k^t indicates the importance of parameter θ_k^t. In MAS [49], ω_k is used to regularize the MAS loss function. Here, similar loss function as used in MAS is defined and regularized and is named the regularized loss function L_R for updating the target model C on $\mathcal{M}$. As such, for the set of features $\mathcal{X}$ and their related output masks $\mathcal{Y}$ in memory $\mathcal{M}^t$, and given a target model C^{t-1} with K parameters Θ^{t-1}, the regularized loss function L_R is defined as

$$L_R(\Theta^t, \mathcal{M}^t) = L(\Theta^t, \mathcal{M}^t) + \gamma \sum_{k=1}^{K} \omega_k^{t-1} \big(\theta_k^t - \theta_k^{t-1}\big)^2, \tag{4.2}$$

where $L(\Theta^t, \mathcal{M}^t)$ is as described in Equation (3.2). The latter term is the MAS regularization, controlled by γ which is set by cross-validation, and t is the time index. This regularization term adds a focus on prior $p(\Theta^{t-1}|\mathbb{D}^{t-1})$ by involving the learned information Θ^{t-1} at the time of updating the target model's parameters Θ^t. The overall goal of the MAS solution is that the loss L_R allows the target model to be updated while preserving its most important previously learned knowledge in Θ^{t-1}.

The effectiveness of the loss function L_R for MAS deteriorates over time (frames) as $\Omega^t = \{\omega_k^t\}_{k=1}^K$ loses its effectiveness in regularization, since most parameters become more important as the number of tasks (target model updates) is increased. In other words, in many target model update steps, the memory $\mathcal{M}$ would be different in different time step t, and samples in the memory also would have different temporal weights d. Consequently, different gradient magnitudes would be added to Ω^t and all $\omega^t \in \Omega^t$ would be a big number, indicating the same importance of other parameters. On the other hand, EWC [5] keeps gradients $\{u_k^t\}_{k=1}^K$ and the parameters $\{\theta_k^t\}_{k=1}^K$ of each update step t for regularizing the loss function in the Equation (3.2); however, this solution is also not efficient for Online VOS since it requires to store the parameters of the target model Θ after each update step. In other words, the proposed solution by EWC [5] requires keeping a copy of the target model's

parameters alongside their importance indicators in a memory. Given the drawbacks of EWC and MAS, the Gated-Regularization Continual Learning (GRCL) proposed in the following section aims to address those drawbacks and improve the performance of Online VOS from a continual learning perspective.

4.2 Gated-Regularizer Continual Learning

In Section 4.1, limitations of two conventional continual learning methods (MAS [49] and EWC [5]) are discussed, in the context of being used directly in an Online VOS framework. The comparison between the proposed prior-focused approach and MAS will be demonstrated in Section 7.4. In this section, a proposed Gated Regularized Continual Learning (GRCL) is introduced that tackles the conventional continual learning methods on Online VOS. The proposed GRCL addresses the limitations of MAS and EWC:

- GRCL uses a binary data structure (map) as the importance parameter indicator, which is more feasible than EWS's to store in memory. It targets and solves the EWC memory limitation problem.
- GRCL is inspired by MAS and does not need to store the target model's parameters after each update step. It also addresses the EWC memory requirement problem.
- Considering the speed limitations of some Online VOS methods, GRCL considers each binary importance parameter as a gating function, which is more effective than LWL and MAS for training a model over a small number of epochs.
- Another mechanism that is used in GRCL is a dynamic gated-regularizer memory, which is expanded and shrunk based on the target model's regularized parameter and keeps the target model effective during evaluation.

GRCL uses a binary regularizer that acts like a gating function that lets a parameter θ be updated or frozen and fixed. This gated regularizer has some benefits such as smaller memory requirement and more efficiency on a target model C which is supposed to be updated in some small number of epochs.

Here, similar to prior-focused approaches in continual learning (MAS and LWL), the regularizer is associated with the parameter importance and is calculated based on the magnitude of the gradients found during the back-propagation in each updating step.

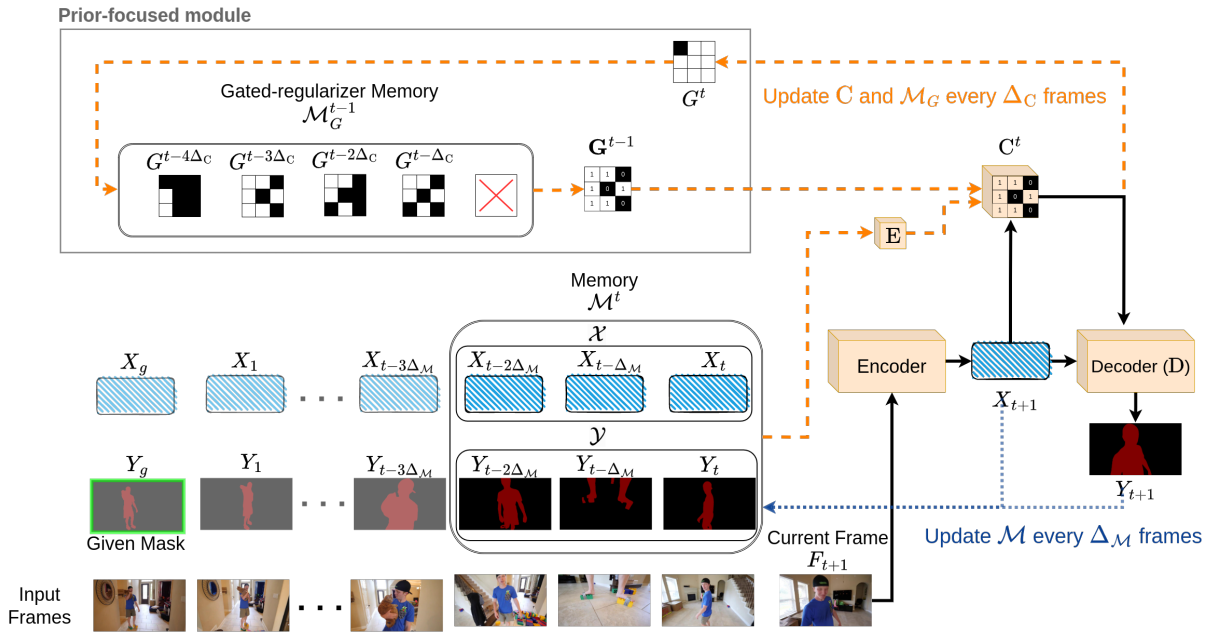

Figure 4.1: The proposed Online VOS framework, with adopted Gated-Regularized Continual Learning (GRCL): At time t, the overall gated-regularizer map $\mathbf{G}^{t-1}$ is calculated using the stored gated maps in the gated-regularizer memory $\mathcal{M}_G^{t-1}$ and regularizes the process of updating C^{t-1} on $\mathcal{M}^t$. Finally, after updating C^{t-1} to C^t, $\mathcal{M}_G^{t-1}$ is updated using the calculated G^t from C^t. The frame images used in the figure are taken from the Long Video dataset [1].

Thus, GRCL is formulated such that, instead of accumulating the importance parameters in Ω^t as was suggested in Section 4.1, it stores a maximum limited number of P binarized gated maps $\{G^j\}_{j=1}^{|\mathcal{M}_G^t|}$ in a gated-regularizer memory $\mathcal{M}_G^t$, where the size of $\mathcal{M}_G^t$ is limited $\left(\left|\mathcal{M}_G^t\right| \leq P\right)$. Each gated-regularized map G^t contains the information about the important parameter of updating step t. It is worth mentioning that G^t has the same dimension of the target model's parameters Θ but in binary form . To calculate G^t, after accumulating the magnitude of the gradients in $U^t = \{u_k^t\}_{k=1}^K$, an element of a binary gated-regularizer map $g_k^t \in G^t$ will be defined as

$$g_k^t = \begin{cases} 1 & \text{if } \frac{u_k^t}{\max_k(U^t)} > h \\ 0 & \text{else} \end{cases} \tag{4.3}$$

where $0 < h < 1$ is a threshold which is determined based on the distribution of the gradients in U^t. The threshold h is mainly dependent on the size of the target model CC. To set the value for h, U^t is normalized by the maximum gradient stored $\max_k(U^t)$ and the n^{th} percentile of the normalized distribution would define the selected value for the threshold h which is selected using cross-validation and would be different for different target model sizes and the updating process. The bigger the value of h, the more sparse the resulting gated-regularized map G^t.

For regularization, we need to benefit from all G in $\mathcal{M}_G$ and the overall gated-regularized map $\mathbf{G}$ is introduced. Thus, at each update step t, an overall gated-regularized map $\mathbf{G}^{t-1}$ is defined from all stored gated-regularized maps $\{G^j\}_{j=1}^{\left|\mathcal{M}_G^{t-1}\right|}$ in the memory $\mathcal{M}_G^{t-1}$ as

$$\mathbf{G}^{t-1} = \bigvee_{j=1}^{J} G^j \quad , \quad J = \left|\mathcal{M}_G^{t-1}\right| . \tag{4.4}$$

Here $\bigvee$ is the "Logical Or" operator, and $\left|\mathcal{M}_G^{t-1}\right|$ is the number of occupied memory cells in $\mathcal{M}_G^{t-1}$. For example, in Figure 4.1, $\left|\mathcal{M}_G^{t-1}\right|$ is 4. Given the current overall gated-regularizer map $\mathbf{G}^{t-1}$, the gated-regularized loss function L_G can be formulated as

$$L_G(\Theta^t, \mathcal{M}^t, \mathbf{G}^{t-1}) = L(\Theta^t, \mathcal{M}^t) + \gamma \sum_{k=1}^{K} \mathbf{g}_k^{t-1} \left(\theta_k^t - \theta_k^{t-1}\right)^2, \tag{4.5}$$

where $\mathbf{g}_k^{t-1} \in \mathbf{G}^{t-1}$ is a binary element of the overall gated regularizer map $\mathbf{G}^{t-1} = \{\mathbf{g}_k^{t-1}\}_{k=1}^{K}$ and it is different from g in Equation (4.3), each element of each gated-regularizer map G in the gated-regularizer memory $\mathcal{M}_G$. With a large enough coefficient γ, the binary regularization term acts as a gating function that allows some parameters to be updated and others to be frozen. After updating the target model C^{t-1} and creating C^t is the time to determine the important parameter of the current update and make sure they are preserved for some further updates. Thus, a new gated-regularizer map (G^t) is defined and memory $\mathcal{M}_G^{t-1}$ is updated by (G^t) to form $\mathcal{M}_G^t$. It is worth mentioning that $\mathcal{M}_G^{t-1}$ would be updated if C^t was updated, thus Δ_{C} specifies the updating step size for $\mathcal{M}_G$ as well.

Figure 4.1 shows an Online VOS framework at time t when the target model C^t is regularized by the proposed GRCL. Algorithm 2 described the steps of GRCL, where updating $\mathcal{M}_G^{t-1}$ to $\mathcal{M}_G^t$ happens after updating the target model C^{t-1} to C^t (line 5 and 6 in Algorithm 2). One of the main advantages in formulating the prior-focused loss function of the Online VOS framework as L_G, is to store an efficient set of binary maps $\{G^j\}_{j=1}^{|\mathcal{M}_G|}$ in $\mathcal{M}_G^t$ which is much smaller in size compared to the sets of features $\mathcal{X}$ and masks $\mathcal{Y}$ stored

in $\mathcal{M}^t$. For example, in computers, a single bit can be either 0 or 1 and it can be used for storing binary data, while a "double" data type (which provides greater precision than a "float") is typically eight bytes (64 bits) in size. thus, a data unit of a binary file is 64 time smaller that the double data unit type. A quantitative comparison between $\mathcal{M}$ and $\mathcal{M}_G$ is done in Section 7.4.4.

It is worth noting that the feature encoder, decoder D, and label encoder network E in the proposed architecture all trained offline, and the same trained models were used in all experiments in Chapter 7. Additionally, the memory is initialized by the encoded features of the given frame F_g with the provided ground-truth mask Y_g, as defined in semi-supervised VOS frameworks.

4.2.1 Dynamic Gated-Regularizer Memory

The gated-regularizer memory $\mathcal{M}_G^t$ can have a fixed size of P similar to $\mathcal{M}^t$ that has a fixed size of N; however, as the number of stored gated-regularized maps is increased, the degrees of freedom of the target model C^t for learning new information in the memory will be decreased, and that could have negative effects on the performance of the model. To handle this problem, In this section, a dynamic mechanism is proposed to make the gated-regularizer memory $\mathcal{M}_G^t$ of GRCL dynamic in size. To do this, when the overall gated-regularized map $\mathbf{G}^{t-1}$ is calculated, the number of ones in $\mathbf{G}^{t-1}$ determines the number of regularized parameters of the target model, and if it is smaller than a certain threshold, GRCL tends to expand $\mathcal{M}_G^t$. On the other hand, if the number of ones in $\mathbf{G}^{t-1}$ is greater than another threshold, $\mathbf{G}^{t-1}$ will be shrunk, and the oldest stored gated-regularized maps in the memory $\mathcal{M}_G^{t-1}$ will be removed from memory to keep the number of regularized parameters below and above certain thresholds.

The number of regularized parameters upper bound threshold η_u is proportionate to the number of target model parameters K, thus, the upper-bound of P is when the number of regularized parameters (ones in $\mathbf{G}^{t-1}$) reaches $\eta_u = \xi_u \times K$ and the lower-bound of P is when the number of regularized parameters be less than $\eta_l = \xi_l \times K$. The two ξ_u and ξ_l ratios would be found for each target model and number of training epochs using cross-validation.

Thus, GRCL does not need to make any changes on Online VOS methods. It only needs to regularize the target model C updating loss function. Additionally, the hyper-parameters for GRCL that should be tuned are h that is used to binarize the gated maps (G) and also two other ratios (ξ_l and ξ_u) which determine the the lower bound and upper bound of regularized parameters (η_u and η_l) in C and make the $\mathcal{M}_G$ dynamic in size.

Algorithm 2: Proposed GRCL solution.

Input: $\{X_{t+1}, \mathcal{M}^t, \mathcal{M}_G^{t-1}, \Theta^{t-1}, \mathrm{D}, \mathrm{E}, \Delta_\mathrm{C}, \Delta_\mathcal{M}, t_\mathrm{C}, t_\mathcal{M}\}$

// Updating the target model C^{t-1}.

if $t - t_\mathrm{C} = \Delta_\mathrm{C}$ **then**

  $t_\mathrm{C} \leftarrow t$

  // Creating the overall gated-regularizer map using Equation (4.4).

  $\mathbf{G}^{t-1} \leftarrow$ Equation 4.4

  // Updating the target model C^{t-1} parameters using Equation (4.5).

  $\Theta^t \leftarrow L_G(\Theta^{t-1}, \mathcal{M}^t, \mathbf{G}^{t-1})$

  // Creating a new gated-regularizer map from C^t.

  $G^t \leftarrow \Theta^t$

  // Updating the gated-regularizer memory $\mathcal{M}_G^{t-1}$.

  $\mathcal{M}_G^t \leftarrow \mathcal{M}_G^{t-1} \cup G^t$

else

  $\Theta^t \leftarrow \Theta^{t-1}$

  $\mathcal{M}_G^t \leftarrow \mathcal{M}_G^{t-1}$

end

// Segmenting the object using the decoder D.

$Y_{t+1} \leftarrow \mathrm{D}\Big(X_{t+1}, \mathrm{E}\big(\mathrm{C}^t(X_{t+1})\big)\Big)$

// Updating the memory $\mathcal{M}^t$.

if $t - t_\mathcal{M} = \Delta_\mathcal{M}$ **then**

  $t_\mathcal{M} \leftarrow t$

  $\mathcal{M}^{t+1} \leftarrow \mathcal{M}^t \cup \{X_{t+1}, Y_{t+1}\}$

else

  $\mathcal{M}^{t+1} \leftarrow \mathcal{M}^t$

end

return $\{Y_{t+1}, \mathcal{M}^{t+1}, \mathcal{M}_G^t, \Theta^t, t_\mathrm{C}, t_\mathcal{M}\}$

Chapter 5

Likelihood-focused VOS solution

Likelihood-focused approaches [46, 54] are used in continual learning to benefit from a subset of data that is used in the training of preceding tasks in order to keep their performance reasonable on previously trained tasks. In other words, a likelihood-focused approach tries to provide the best data that the Maximum Likelihood Estimation (MLE) approach could provide for the most generalized model. One challenge for these approaches is to provide the most compact and informative subset of previously used data in the training process of the current task. It is worth noting that in this chapter, similar to Chapter 4, we still maximize the Bayesian formulation of the posterior probability, which is defined in Equation (3.6). In other words, the focus the proposed approach in this chapter is on likelihood part $p\big(\mathcal{M}^t|\Theta\big)$ where $\mathcal{M}^t$ is the memory and Θ is the set of the target model's parameters; however, the prior is still implicitly involved in the optimization of Equation (3.5). The fine-tuning of target model parameters, which implicitly contains some past information from previous update steps, is an example of involving prior information in a likelihood-focused approach. Likelihood-focused methods try to have a more compact and efficient subset of available data, which is $\mathcal{M}^t$ in the general online Video Object Segmentation (VOS) framework. By minimizing $p\big(\Theta|\mathcal{M}^t\big)$, we are solving a Maximum A Posteriori (MAP) problem, but the question here is how we can focus on the likelihood part of the Bayesian formulation of an Online VOS by providing a compact and efficient subset of data. Section 5.1 discusses what it means by an "efficient" subset of data by defining the limitations of the memory $\mathcal{M}^t$ in an Online VOS framework. In Section 5.2, the proposed likelihood-focused method is introduced.

5.1 Memory limitation

In online VOS, each updating step t is done on memory $\mathcal{M}^t$ and this memory is different from one updating step to another updating step. If a sudden appearance drift happens on the object, the model cannot handle this drift even though it has already been trained on the similar object appearance on preceding memory updates, and this is the same as catastrophic forgetting of network models in continual learning. Given this forgetting behaviour of an Online VOS, a trivial solution for mitigating this problem is simply to have an unlimited memory size and keep all already visited information in the memory $\mathcal{M}$. However, there are some problems with this idea for an Online VOS:

1. The bottleneck of an Online VOS method is the target model C updating computational complexity. This complexity is related to the size of target model, the number of training samples, and the number of training epochs. For a long video with an unlimited memory size $\mathcal{M}$, the computational complexity of updating the target model would be increased as the memory grows over time. Thus, the the memory size should be limited to handle this problem.

2. It is difficult for a limited-size target model C to extract generalized discriminating information from a large memory $\mathcal{M}$ with a gigantic set of highly variable and unbalanced data. Here "unbalanced" refers to the situation that many similar frames are placed continuously in a video sequence, and their features and estimated masks will be placed in the memory $\mathcal{M}$; however, there are a small number of related and useful samples in the memory for segmenting the current frame. As such, the effectiveness of updating the target model C^t on long videos deteriorates even without considering the computational complexity.

3. Another limitation of Online VOS is determining the importance of samples for the current update of target model C in which d_n is used as the temporal weight for each sample in the memory and is used in the loss function of Equation (3.2). The temporal weight is a deterministic weight decay coefficient that is larger for the most recent sample in the memory $\mathcal{M}^t$ and smaller for the oldest evaluated sample in the memory. This weighting mechanism will cause the target model $\mathcal{M}$ to forget about old samples in its memory. An old sample in the memory could be forgotten, even if it is very helpful for segmenting the current frame.

To tackle these limitations, the proposed likelihood-focused approach in this chapter, benefits from a dynamic working memory $\mathcal{M}_W$ which is a subset of $\mathcal{M}$, and is used for updating the target model. This new approach addresses the mentioned problems:

i. Allowing a limited-size target model to benefit from a large memory, and

ii. The update step becomes significantly more efficient, since it is training on a smaller working memory $\mathcal{M}_W^t$.

iii. All of the samples in the memory could have a considerable weight in the training loss function of target model independent of their temporal weight d_n, by re-weighting the selected samples from $\mathcal{M}$.

To form the dynamic working memory $\mathcal{M}_W^t$ at time t, we need a memory selection method that selects a subset of $\mathcal{M}^t$ and this method should have the following properties:

1. It should be fast enough to not have negative effects on the efficiency of the Online VOS framework.

2. It should output a diverse working memory $\mathcal{M}_W^t$ to handle the drift that may happen between current target model update and next update.

3. The proposed memory selection method should output a sufficiently small working memory $\mathcal{M}_W^t$ in comparison to $\mathcal{M}^t$.

4. It should also provide a weight for each selected sample in $\mathcal{M}_W^t$ indicating the importance of the selected sample for the current update.

The proposed Reconstruction-based Memory Selection Continual Learning (RMSCL) is a fast and efficient Online VOS approach which benefits from a diverse working memory and is introduced in Section 5.2.

5.2 Reconstruction-based Memory Selection Continual Learning

The RMSCL approach proposed in this section adapts a methodology similar to those of likelihood-based (rehearsal) approaches in continual learning, where a set of selected observations from the preceding tasks is preserved in a so-called working memory $\mathcal{M}_W^t$ to mitigate the catastrophic forgetting by the online model of proceeding tasks.

RMSCL does not change the memory update mechanism of Online VOS; instead, it offers a memory selection mechanism to select diverse samples from the memory $\mathcal{M}$. In

general, likelihood-focused methods make an effort to keep the memory diverse enough with the samples that come from the tasks and domains that came before them. To put it another way, in the hypothetical situation that a drift happens in the video sequence and the memory $\mathcal{M}$ does not contain any useful information related to the current drift, RMSCL could possibly experience some difficulty handling that drift.

As was discussed in Section 5.1, $\mathcal{M}_W^t$ needs to be a small diverse memory which contains the required features and masks of preceding evaluated frames. Thus, the goal of the proposed RMSCL is to select some pairs of samples from memory $\mathcal{M}^t$ and place them in $\mathcal{M}_W^t$ to be used for updating target model C^{t-1} at time t. This memory selection is performed on $\mathcal{M}^t$ every Δ_{C}. The selection of samples from memory is formulated as a LASSO [94] optimization problem: To update the target model C^{t-1}, the optimal linear reconstruction of the stored features $\mathcal{X} \in \mathcal{M}^t$ for the next feature X_{t+1} is identified via a L_{RMSCL} constraint on a randomly initialized vector of coefficients Ψ by minimizing

$$\Psi^t = \arg\min_{\Psi} L_{RMSCL}(\Psi, \mathcal{M}^t, X_{t+1}) = \arg\min_{\Psi} \left(\frac{1}{2} \left\| X_{t+1} - \Psi\mathcal{X} \right\|_2^2 + \lambda \|\Psi\|_1 \right). \tag{5.1}$$

It is worth noting that updating the target model C^{t-1} and creating C^t will happen before segmenting the object in frame F_{t+1} and predicting Y_{t+1} using the updated C^t at time t. Moreover, $\mathcal{X}$ contains $\left|\mathcal{M}^t\right|$ features ($\mathcal{X} = \{X_l\}_{l=1}^{\left|\mathcal{M}^t\right|}$), similarly Ψ consists of $\left|\mathcal{M}^t\right|$ coefficients ($\Psi = \{\psi_l\}_{l=1}^{\left|\mathcal{M}^t\right|}$) weighting each feature X_l in reconstructing of X_{t+1}. In other words, we want to have the best sparse linear reconstruction of new received frame X_{t+1} using the stored features $\mathcal{X}$ in memory $\mathcal{M}^t$.

The L_{RMSCL} loss leads to a sparse set of coefficients because of L_1-norm in Equation (5.1) [95], meaning that only a small number of coefficients Ψ are non-zero after the optimization process, and the positive coefficients ψ and their associated features X are selected and are placed in $\mathcal{M}_W^t$ for updating target model C^{t-1}.

It is important to mention that the deterministic temporal weight d_n is not involved in L_{RMSCL} loss function in Equation (5.1) and instead RMSCL re-weights the selected samples in $\mathcal{M}_W^t$ by the coefficient Ψ calculated in Equation (5.1). This re-weighting enables RMSCL to include the significance of selected samples in the current update phase. Thus, d_n is replaced with ψ_n in Equation (3.2) as:

$$L(\Theta^t, \mathcal{M}_W^t) = \sum_{n=1}^{\left|\mathcal{M}_W^t\right|} \left\| \psi_n W_n \Big(\mathrm{E}(Y_n) - \mathrm{E}\big(\mathrm{C}^t(X_n)\big) \Big) \right\|_2^2 + \sum_{k=1}^{K} \lambda \, {\theta_k^t}^2. \tag{5.2}$$

Algorithm 3: Proposed RMSCL solution.

Input: $\{X_{t+1}, \mathcal{M}^t, \Theta^{t-1}, \mathrm{D}, \mathrm{E}, \Delta_\mathrm{C}, \Delta_\mathcal{M}, t_\mathrm{C}, t_\mathcal{M}\}$

// Updating the target model C^{t-1}.

if $t - t_\mathrm{C} = \Delta_\mathrm{C}$ **then**

  $t_\mathrm{C} \leftarrow t$

  // Creating the working memory by optimizing Equation (5.3).

  $\mathcal{M}_W^t \leftarrow L_{RMSCL}(\Psi, \mathcal{M}^t, X_{t+1})$

  // Updating the target model C^{t-1} parameters using Equation (3.2).

  $\Theta^t \leftarrow L(\Theta^{t-1}, \mathcal{M}_W^t)$

else

  $\Theta^t \leftarrow \Theta^{t-1}$

end

// Segmenting the object using the decoder D.

$Y_{t+1} \leftarrow \mathrm{D}\Big(X_{t+1}, \mathrm{E}\big(\mathrm{C}^t(X_{t+1})\big)\Big)$

// Updating the memory $\mathcal{M}^t$.

if $t - t_\mathcal{M} = \Delta_\mathcal{M}$ **then**

  $t_\mathcal{M} \leftarrow t$

  $\mathcal{M}^{t+1} \leftarrow \mathcal{M}^t \cup \{X_{t+1}, Y_{t+1}\}$

else

  $\mathcal{M}^{t+1} \leftarrow \mathcal{M}^t$

end

return $\{Y_{t+1}, \mathcal{M}^{t+1}, \Theta^t, t_\mathrm{C}, t_\mathcal{M}\}$

Here, $\big|\mathcal{M}_W^t\big|$ is the size of dynamic working memory $\mathcal{M}_W^t$, equal to number of non-zero positive $\{\psi_n\}$. The only problem with the LASSO minimization of Equation (5.1) is that its computational complexity depends on the dimensionality of feature X, such that a gigantic feature size can lead optimizing Equation (5.1) to becoming the bottleneck of Online VOS. In order to handle this problem, a channel based max pooling function pool$(\cdot)$ is applied on each feature X, such that Equation (5.1) becomes

$$\begin{aligned}\Psi^t &= \arg\min_{\Psi} L_{RMSCL}(\Psi, \mathcal{M}^t, X_{t+1}) \\ &\simeq \arg\min_{\Psi} \left(\frac{1}{2}\big\|\mathrm{pool}(X_{t+1}) - \Psi\mathrm{pool}(\mathcal{X})\big\|_2^2 + \lambda\|\Psi\|_1 \quad \text{s.t. } \Psi \geq 0\right). \end{aligned} \tag{5.3}$$

The pooling function pools the feature X with dimension of $C \times W \times H$ to $1 \times W \times H$ by pooling over channels C. In other words, the pooling function acts like a dimensionality

reduction function which makes the input data be C times smaller in size. It is worth noting that the pooling function is only performed for estimating the coefficient set Ψ; it is still the actual features $\mathcal{X}$ which are used for creating the working memory $\mathcal{M}_W^t$ and updating the target model. Moreover, a constrain is used in Equation (5.3) that force Ψ to be not negative.

Figure 5.1 shows an Online VOS pipeline resulting from the proposed RMSCL. The flow diagram of RMSCL is shown in Algorithm 3, where the order of updating memories and target model are shown. Moreover, two update time indices $t_{\mathcal{M}}, t_{\mathrm{C}}$ for memory and target model are updated alongside the memory $\mathcal{M}$ and target model C.

The proposed RMSCL method has two main contributions: first, it makes updating the target model C on a small number of samples in $\mathcal{M}_W^t$ which has more efficient computational complexity in comparison to updating the target model on all of the samples in $\mathcal{M}^t$. The second contribution is to re-weight the selected samples in $\mathcal{M}_W^t$ by considering Ψ as the set of sample weights for selected samples from memory $\mathcal{M}^t$. The re-weighting of selected old samples from $\mathcal{M}^t$ addresses the problem of ignoring (forgetting) information in the memory $\mathcal{M}^t$ which would have a small temporal weight d_n which is shown in Equation (3.2). The reconstruction-based sample selection and the re-weighting idea place the proposed RMSCL in the category of likelihood-focused solutions for Online VOS.

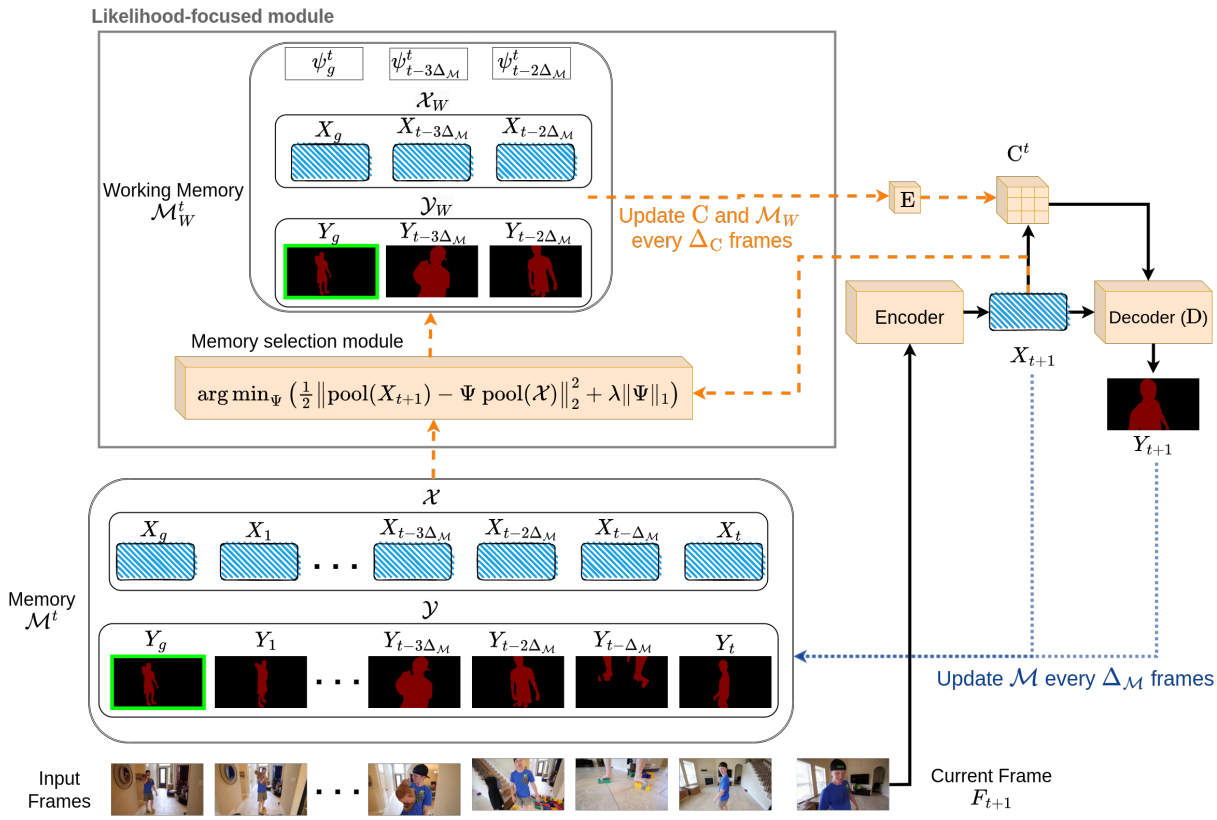

Figure 5.1: The proposed Online VOS framework with augmented Reconstruction-based Memory Selection Continual Learning (RMSCL). At time t, three samples associated to three positive ψ are selected using a reconstruction based (LASSO) optimization. The frame images used in the figure are taken from Long Video Dataset [1].

Chapter 6

Posterior-focused VOS solution

Bayesian inference [96] computes the posterior probability based on the likelihood and prior probabilities defined in Bayes' theorem. In Bayesian inference, we are looking for the posterior probability. Chapter 4 and 5 focus on prior and likelihood; however, the proposed method in this chapter focuses on the end results or outputs $p(\Theta|\mathcal{M})$ to find the best solution given all available information, including data in memory $\mathcal{M}$ and prior knowledge in the model parameters Θ. In other words, the proposed posterior-focused VOS solution in this chapter focuses on maximizing all probability distributions of an Online VOS problem formulation, which is defined in Equation (3.5). Focusing on both prior and likelihood at the same time increases the computational complexity of the model in comparison to GRCL and RMCL. Using both RMSCL and GRCL at the same time would have some limitations such as inefficiencies, or, in other words, overlapped effects, between the likelihood-focused (RMSCL) and prior-focused (GRCL) methods. The overlapped effects here mean GRCL and RMSCL have almost the same impact on the Online VOS method. Section 6.1, the proposed Hybrid solution is discussed.

6.1 Hybrid solution

The proposed posterior-focused solution in this chapter is classified as a Hybrid method [56, 97] as is defined as continual learning. Hybrid methods usually benefit from three different continual learning solutions: regularization-based, replay-based, and structural based [56]. Here, structural solutions of continual learning are not used since those models try to expand the model (increasing the parameters of the model) while keeping other important parameters fixed. For an Online VOS solution, expanding the model size over time is not

an option since the bottleneck of an Online VOS is the target model, and the computational complexity of Online VOS would be strongly affected by increasing the size of the target model C. The challenges that a Hybrid method of continual learning aims for are better robustness and generalization in different situations.

The proposed Hybrid solution in this chapter includes GRCL and RMSCL modules, but it also includes a message-passing mechanism between the GRCL and RMSCL components, which is the fundamental contribution of Hybrid. The reason the Hybrid approach need the message -passing module is because GRCL can amplifies the positive and negative effects of RMSCL and we need to find a way to control and handle this problem. GRCL essentially force the model to not forget the preceding update of the target model C on the working memory $\mathcal{M}_W$ of RMSCL.

The message-passing mechanism tells GRCL when to store the gated-regularized map G^t into $\mathcal{M}_G^{t-1}$ after updating the target model C^t on $\mathcal{M}_W^t$. Therefore, it is necessary to have an indicator that demonstrates how usefull is the most recent update of C^t for the purpose of regularizing updates in the future. It is abundantly clear that the quality of an updated C^t has a direct and substantial relationship with the working memory $\mathcal{M}_W^t$ that is utilized. The reconstruction error, represented by r_e^t, of the current input frame's feature X_{t+1} is specified as the indicator for placing G^t in $\mathcal{M}_G^{t-1}$:

$$r_e^t = R(X_{t+1}, \Psi^t, \mathcal{X}_W) = \left\| X_{t+1} - \Psi^t \mathcal{X}_W \right\|_2^2. \tag{6.1}$$

Here, as explained in Equation (5.3), Ψ^t is a vector of coefficients estimated from the minimization of Equation (5.3) in the RMSCL approach. Ψ^t contains some important information. First, the positive elements of Ψ^t determine the selected samples from memory $\mathcal{M}$. Second, the value of each element in Ψ^t indicates the importance of the selected samples in working memory $\mathcal{M}_W$. The positive and non-zero coefficients (Ψ^t) are multiplied by $\mathcal{X}_W$ which are selected features from $\mathcal{M}^t$ and are stored in $\mathcal{M}_W^t$. It is worth noting that $\Psi^t \mathcal{X}_W$ presents a linear reconstruction of current input feature X_{t+1} without applying pooling function; while the positive and non-zero coefficients Ψ^t is calculated using the pooled features as is shown in Equation (5.3). Another indicator that shows the quality of current update is the number of selected samples in $\mathcal{M}_W^t$ which is shown by $\left|\mathcal{M}_W^t\right|$.

After calculating the reconstruction error r_e^t, two thresholds are considered to determine whether $\mathcal{M}_G^{t-1}$ should be updated with G^t. The first proposed threshold used in the Hybrid method is τ, which is set specifically for each adopted Online VOS method. On the other hand, β is the threshold for number of selected samples in $\mathcal{M}_W^t$.

If the associated reconstruction error r_e^t is greater than the threshold ($r_e^t > \tau$) and the number of selected samples in $\mathcal{M}_W^t$ is greater than β (i.e., $\left|\mathcal{M}_W^t\right| > \beta$), then the gated-regularized map G^t is placed in the gated-regularizer memory $\mathcal{M}_G^{t-1}$. The idea behind this

decision-making process is that if the reconstruction error r_e^t is greater than the threshold τ, the current frame's feature X_{t+1} is most likely from a typical frame at the middle of a video sub-chunk (of which examples are shown in Figure 2.4), and we may want to remember the important parameters of C^t associated with G^t of this update for dealing with similar frame features when segmenting future frames. Furthermore, the small reconstruction error occurs on very similar frames with no significant differences or missing objects, and the Hybrid method does not need or want to remember those updates. In addition to the reconstruction error threshold τ, β specifies the number of chosen samples to be utilized to reconstruct the current feature. If the number of selected samples is fewer than a specific threshold β, the new update most likely has selected the most recent frames' features and has not provided useful information compared to the prior update steps. The two thresholds (τ and β) are selected using cross-validation for each baseline, augmented by the Hybrid approach separately.

The Hybrid solution module is shown in Figure 6.1 in a grey border box. The green dotted arrows shows the message-passing part of Hybrid. The proposed Hybrid solution, with the explained message-passing approach, has an algorithmic structure that is defined in Algorithm 4. The proposed Hybrid algorithm has three main parts:

1. The likelihood-focused (RMSCL) part that generates the working memory $\mathcal{M}_W^t$ by optimizing Equation (5.3). Hybrid does not need to tune the hyper parameters of

2. The prior-focused (GRCL) part that calculates the overall gated regularizer map $\mathbf{G}^{t-1}$ from the gated-regularizer memory $\mathcal{M}_G^{t-1}$ to the target model's updating loss function in Equation (4.5).

3. The message-passing mechanism that decides whether $\mathcal{M}_G^{t-1}$ should be updated by G^t calculated from the updated target model on $\mathcal{M}_W^t$.

The loss function L_H that is used in the proposed posterior-focused solution is

$$L_H(\Theta^t, \mathcal{M}_W^t, \mathbf{G}^{t-1}) = L(\Theta^t, \mathcal{M}_W^t) + \gamma \sum_{k=1}^{K} \mathbf{g}_k^t \big(\theta_k^t - \theta_k^{t-1}\big)^2. \tag{6.2}$$

The only difference between Equation (6.2) and Equation (4.5) is that in Equation (6.2), a working memory $\mathcal{M}_W^t$ is used that is smaller than $\mathcal{M}^t$; thus the proposed Hybrid solution could be faster than the proposed GRCL method with a large memory $\mathcal{M}^t$. It is worth noting that for doing the fusion between the RMSCL and GRCL methods, we first need to apply the RMSCL method and then update the target model's parameters using the

gated regularizer and after that, the message-passing mechanism of Hybrid decides whether the gated-regularizer memory $\mathcal{M}_G^{t-1}$ should be updated. Algorithm 4 demonstrates the proposed Hybrid method in detail. The order of performing different parts of the proposed Hybrid method is shown in Algorithm 4. The contribution of the proposed Hybrid approach is that it gets the likelihood (RMSCL) and prior (GRCL) methods with the specified hyper-parameters and does a smart fusion over those two methods to achieve the best performance in terms of accuracy and robustness.

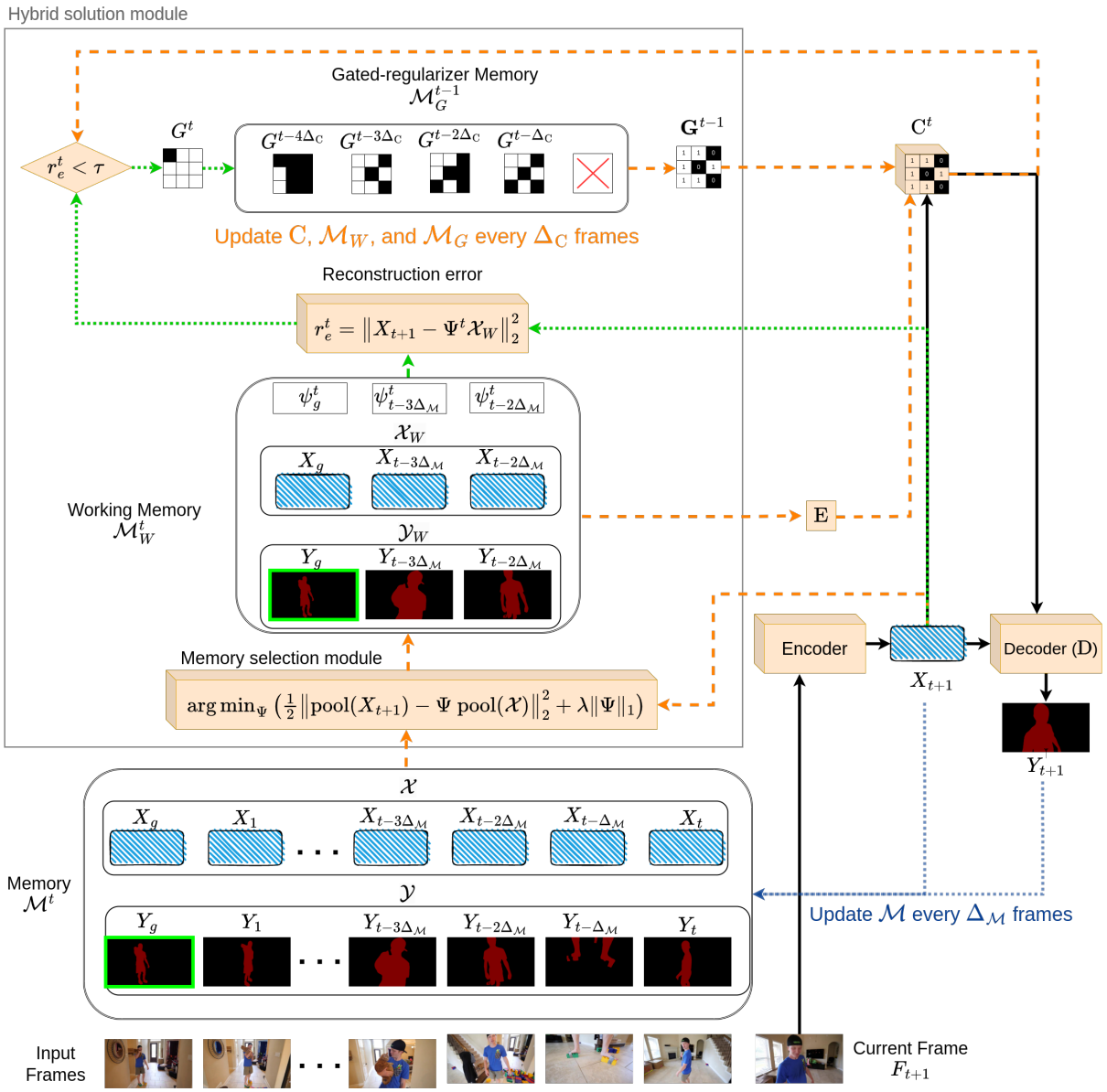

Figure 6.1: The proposed Hybrid Online VOS framework. The green dotted arrows shows the message-passing mechanism discussed in Section 6.1. The Hybrid module is shown in a grey rectangle and augments the Online VOS framework. The frame images used in the figure are taken from Long Video Dataset [1]

Algorithm 4: Proposed Hybrid solution.

Input: $\{X_{t+1}, \mathcal{M}^t, \mathcal{M}_G^{t-1}, \Theta^{t-1}, \mathrm{D}, \mathrm{E}, \Delta_{\mathrm{C}}, \tau, \beta, \Delta_{\mathcal{M}}, t_{\mathrm{C}}, t_{\mathcal{M}}\}$

// Updating the target model C^{t-1}.

if $t - t_{\mathrm{C}} = \Delta_{\mathrm{C}}$ **then**

  $t_{\mathrm{C}} \leftarrow t$ // Creating the working memory by optimizing Equation (5.3).

  $\mathcal{M}_W^t, \Psi^t \leftarrow L_{RMCL}(\Psi, \mathcal{M}^t, X_{t+1})$

  // Creating the overall gated-regularizer map using Equation (4.4).

  $\mathbf{G}^{t-1} \leftarrow$ Equation (4.4)

  // Updating the target model C^{t-1} parameters using Equation (6.2)

  . $\Theta^t \leftarrow L_H(\Theta^{t-1}, \mathcal{M}_W^t, \mathbf{G}^{t-1})$

else

  $\Theta^t \leftarrow \Theta^{t-1}$

end

// Calculating the reconstruction error using Equation (6.1)

$r_e^t \leftarrow R(X_{t+1}, \Psi^t, \mathcal{X}_W)$

// Updating the gated-regularizer memory $\mathcal{M}_G^{t-1}$.

if $r_e^t > \tau$ *and* $\left|\mathcal{M}_W^t\right| > \beta$ **then**

  // Creating a new gated-regularizer map from C^t.

  $G^j \leftarrow \Theta^t$

  // Updating the gated-regularizer memory $\mathcal{M}_G^{t-1}$.

  $\mathcal{M}_G^t \leftarrow \mathcal{M}_G^{t-1} \cup G^j$

else

  $\mathcal{M}_G^t \leftarrow \mathcal{M}_G^{t-1}$

end

// Segmenting the object using the decoder D.

$Y_{t+1} \leftarrow \mathrm{D}\Big(X_{t+1}, \mathrm{E}\big(\mathrm{C}^t(X_{t+1})\big)\Big)$

// Updating the memory $\mathcal{M}^t$.

if $t - t_{\mathcal{M}} = \Delta_{\mathcal{M}}$ **then**

  $t_{\mathcal{M}} \leftarrow t$

  $\mathcal{M}^{t+1} \leftarrow \mathcal{M}^t \cup \{X_{t+1}, Y_{t+1}\}$

else

  $\mathcal{M}^{t+1} \leftarrow \mathcal{M}^t$

end

return $\{Y_{t+1}, \mathcal{M}^{t+1}, \mathcal{M}_G^t, \Theta^t, t_{\mathrm{C}}, t_{\mathcal{M}}\}$

Chapter 7

Experimental Results

In this chapter, the efficacy of the proposed methods to improve the performance of Online VOS is evaluated by augmenting and assessing state-of-the-art Online VOS frameworks. It is worth noting that all of the proposed solutions, such as prior-focused gated-regularizer continual learning (GRCL), likelihood-focused reconstruction-based memory selection continual learning (RMSCL), and the posterior-focused Hybrid approach, can augment any given Online VOS framework. In Section 7.1, two well-known and state-of-the-art Online VOS frameworks that are used and adopted in this thesis as baseline methods are explained.

7.1 Baseline Online VOS frameworks

To show the impact of the proposed methods of this thesis, two Online VOS methods are chosen. The adopted baseline methods are listed and explained.
LWL [16] is an extension over the well-known FRTM [22] framework, benefiting from a label encoder network E which tells its target model C what to learn [16]. LWL follows the framework structure that is explained in Figure 3.1. In LWL, the encoder, decoder D and label encoder E are trained offline, thus we do not make any changes to them in using the proposed solutions.
JOINT [17] approaches the VOS problem by using an online learning induction branch jointly with a transduction branch, which benefits from a lightweight transformer for providing sufficient temporal and spatial attention to its decoder. In other words, JOINT has all of the parts explained in Figure 3.1 in addition to the transduction part. JOINT has its own trained encoder and decoder D models different from LWL. Moreover, JOINT has

reported the state-of-the-art Online VOS in terms of accuracy. Both methods are mainly designed and explained for segmenting a single object in each frame of a video; however, as is mentioned in the LWL and JOINT papers [16, 17] to work on multiple objects, their method processes each object in each frame independently and blends the predicted masks using the soft-aggregation function [98] over each frame.

The mentioned baselines are augmented by the proposed methods of Chapters 4, 5, and 6 on the datasets that are introduced in the next section.

7.2 Datasets

The proposed solutions are compared for two different types of video sequences: long and short. The Long Videos dataset [1] contains objects with a long trajectory with multiple distribution drifts; the short videos are from the standard DAVIS16 [86], DAVIS17 [87], and YouTube-VOS18 [88] datasets, where the target objects are being tracked in a short period of time and usually without significant abrupt changes in appearance. Evaluating the competing methods on both long and short video datasets demonstrates the robustness of the evaluated methods in different environments.

The Long Videos dataset [1] contains three videos with a single object that are recorded for more than 7000 frames. The target objects could have some sudden appearance changes, which lead to significant representation drifts of the video objects. Each of the videos in the dataset has 21 labelled frames for the evaluation.

With regards to short video datasets, the DAVIS16 [86] validation set has 20 videos, each of which has a single object for segmentation; the validation set of DAVIS17 [87] contains 30 video sequences with multiple objects to be segmented in each frame. The validation set of YouTube-VOS18 has 474 video sequences of 65 seen (which are present in the training set) and 26 unseen object classes. The target objects in short video datasets have mostly a short trajectory, with modest changes in object appearance.

7.3 Experimental setup

In this section, the default setup of the selected methods (LWL and JOINT) and the evaluation scenarios are discussed. A fixed and default setup specified for each baseline is used, with a memory $\mathcal{M}$ with maximum memory sizes of $N = 32$ for LWL and $N = 20$ for JOINT, as suggested in their original publications [16, 17] on the evaluated datasets.

Each unit of memory $\mathcal{M}$ includes a pair of feature and mask label (X and Y). For all experiments, the target model C is updated for only three epochs in each updating step, and to have a fair comparison with the baselines, their suggested hyper-parameters [16, 17] are used for all of the experiments. The target model is updated every time the memory is updated, following the proposed setup in [21].

The memory $\mathcal{M}^0$ is initialized by the feature and the given ground-truth label (X_g, Y_g) of given (ground truth) frame F_g. In all of the experiments, as suggested in semi-supervised Online VOS baselines (LWL and JOINT), the information extracted from F_g is preserved and is used throughout the evaluation of other frames in the video sequence. In the proposed methods, the same concept is followed where in GRCL, the gated-regularizer map G^0 related to the training of the target model C^0 on X_g and Y_g is kept in $\mathcal{M}_G$. For RMSCL, the feature X_g and mask Y_g are always placed in the working memory with a minimum weight ψ_g as shown in Figure 5.1. To maintain the same level of focus on the given frame's information as it is in the baselines, Ψ is normalized in order to keep the same ratio between the maximum element of Ψ and ψ_g. It is worth noting that ψ_g is set to 0.25 for LWL and JOINT based on their original publications [16, 17].

The same pretrained decoder D and encoder models are used for all experiments of LWL, and similarly the same decoder D and encoder models are used for all experiments on JOINT. It is worth to mention that the hyper parameters of the proposed approaches in this thesis are find empirically by using cross-validation and running tests on both short and long video datasets.

To measure the effectiveness of the competing methods, consistent with the standard DAVIS protocol [86], the mean Jaccard $\mathcal{J}$ index, mean boundary $\mathcal{F}$ scores, and the average of $\mathcal{J}\&\mathcal{F}$ are reported for all of the methods. For YouTube-VOS18, the reported results are found using the YouTube-VOS18 official evaluation server [88]. The mean Jaccard $\mathcal{J}$ index, mean boundary $\mathcal{F}$ scores, and the overall score $\mathcal{G}$ of seen and unseen object classes are also reported. The speed of each method is reported on the DAVIS16 dataset [22] in units of Frames Per Second (FPS) when $\Delta_{\mathrm{C}} = \Delta_{\mathcal{M}} = 1$. All experiments were performed using one NVIDIA V100 GPU.

7.4 Experimental results

In this section, the proposed methods are compared and evaluated in different scenarios with baselines and other VOS methods on long and short videos. Before comparing the results with other methods and with different scenarios, we need to investigate the superiority of Online VOS methods over XMem [21] which is the state-of-the-art method on

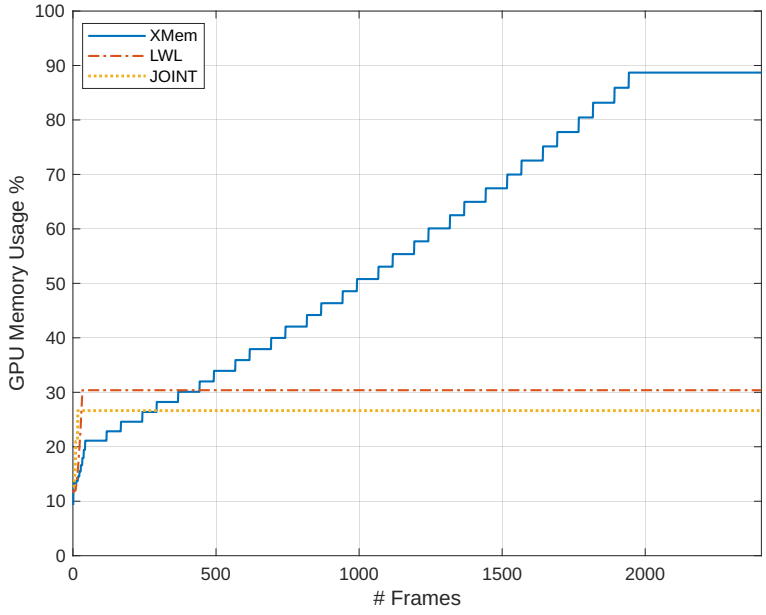

Figure 7.1: GPU memory usage of XMem, LWL and JOINT when processing 2416 frames of the "blueboy" video in the Long Videos dataset [1]. As shown, the GPU memory usage of XMem increases significantly over time, whereas LWL and JOINT have a fixed GPU memory usage.

long video sequences. As is mentioned in XMem [21], but not highlighted, matching-based methods suffer from memory expansion when evaluating long video sequences; however, Online VOS methods are designed to work with a fixed memory size. Figure 7.1 shows the GPU memory usage of LWL, JOINT and XMem on the "blueboy" video sequence from the Long Videos dataset [1]. The Online VOS methods (LWL and JOINT) require only a fixed GPU memory size, which enables them to be used on small devices with more modest GPUs. Figure 7.1 demonstrates that the Online VOS methods do not further increase the GPU memory requirement.

7.4.1 Long Video Evaluation

The effectiveness of the proposed GRCL, RMSCL and Hybrid solutions can be evaluated by augmenting two state-of-the-art Online VOS frameworks, LWL and JOINT; however, the proposed methods can be extended to any Online VOS method having a periodically-

Table 7.1: Comparison results on the Long Videos dataset [1], based on the online VOS baseline methods (LWL and JOINT), their augmented versions with GRCL, RMSCL, Hybrid, and four matching-based VOS methods. The evaluation metric $\mathcal{J}$ is related to the Intersection over Union (IoU) of an estimated object mask and the ground truth, and $\mathcal{F}$ is about how accurate is the boundary.

Method	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$
LWL	79.8 ± 4.2	78.0 ± 4.3	81.6 ± 4.2
LWL-GRCL (proposed)	84.5 ± 1.6	82.8 ± 1.3	86.1 ± 2.0
LWL-RMSCL (proposed)	83.4 ± 2.7	81.5 ± 2.6	85.2 ± 2.8
LWL-Hybrid (proposed)	**86.1 ± 1.4**	84.5 ± 1.4	87.6 ± 1.4
JOINT	67.5 ± 4.4	65.7 ± 4.2	69.3 ± 4.7
JOINT-GRCL (proposed)	70.5 ± 6.8	68.7 ± 6.6	72.3 ± 7.0
JOINT-RMSCL (proposed)	75.6 ± 5.1	73.6 ± 5.0	77.5 ± 5.2
JOINT-Hybrid (proposed)	**76.3 ± 4.6**	74.6 ± 4.3	78.0 ± 4.9
RMNet	59.8 ± 3.9	59.7 ± 8.3	60.0 ± 7.5
STM	80.6 ± 1.3	79.9 ± 0.9	81.3 ± 1.0
STCN	87.3 ± 0.7	85.4 ± 1.1	89.2 ± 1.1
XMem	**89.8 ± 0.2**	88.0 ± 0.2	91.6 ± 0.2

updated target model C, as in Figure 3.1. Table 7.1 shows the results of the selected baselines (LWL and JOINT), each augmented by the proposed GRCL, RMSCL and Hybrid methods, evaluated on the Long Videos dataset [1]. For LWL-GRCL and JOINT-GRCL, the threshold h is dynamically set to the 99.5^{th} percentile of the distribution of normalized U^t in Equation (4.3). Additionally, h is limited ($0.1 < h < 0.55$) for LWL-GRCL and ($0.1 < h < 0.6$) for JOINT-GRCL. Bounding the threshold h prevents the model from regularizing many not important parameters or very few important parameters. The hyper-parameters related to h were selected by cross-validation.

The chosen ratios of GRCL (ξ_l and ξ_u) are 0.07 and 0.15, respectively. These ratios are defined for the target model C and are identical for LWL and JOINT. Taking these two ratios into account, the maximum and lower bounds of number regularized parameters for LWL and JOINT are $\eta_u = 0.15 \times K$ and $\eta_u = \xi_u \times K$, respectively. For LWL, with $K = 73728$, two chosen upper and lower limit thresholds (η_l and η_u) on the number of regularized parameters of the target model C would be 5000 and 11000, respectively. To study the influence of dynamic gated-regularizer memory in GRCL, the effect of employing different constant gated-regularizer memory sizes P in GRCL is discussed in Section 7.5.2.

For the adopted frameworks by RMSCL, the parameter λ defines the sparsity of Ψ in Equation (5.3). To select the best λ, Akaike Information Criterion (AIC) [99, 100] is used for model selection, automatically selecting λ and the number of positive non-zero coefficients Ψ^t, which defines the size of the working memory $\mathcal{M}_W^t$. Thus, for each update step t, in principle $\mathcal{M}_W^t$ could have a different size in comparison to $\mathcal{M}^t$, depending upon the selected λ, the current feature X_{t-1}, and the set features $\mathcal{X}$ in the memory $\mathcal{M}^t$.

It is worth noting that the selected hyper-parameters for Hybrid solution are the sames as the selected parameters for GRCL and RMSCL for each dataset assuming the Hybrid method benefits from the best version of both GRCL and RMSCL and does not need to re-tune new hyper-parameters for its GRCL and RMSCL parts. In addition to the hyper-parameters for GRCL and RMSCL, there are two hyper-parameters (thresholds) for Hybrid which are τ and β. For both LWL and JOINT, $\beta = 2$ which means that in addition to the given ground-truth data which is always placed in the working memory $\mathcal{M}_W^t$, the size of working $\mathcal{M}_W^t$ should also be bigger than $\beta = 2$ in order to update the gated-regularizer memory $\mathcal{M}_G^{t-1}$. Additionally, τ is set to 2000 for LWL and 6000 for JOINT.

For the evaluation of the proposed methods of this thesis, six experiments with six different memory and target model update step sizes $\Delta_{\mathrm{C}} \in \{1, 2, 4, 6, 8, 10\}$ are conducted, where an updated memory $\mathcal{M}^t$ is used to update C^{t-1} to C^t ($\Delta_{\mathcal{M}} = \Delta_{\mathrm{C}}$). The performances of the Online VOS methods vary with different step sizes Δ_{C} and specifically $\Delta_{\mathcal{M}}$, because of the differing distributions that are formed in the memory $\mathcal{M}$ as a function of sampling frequency. This is also induced by looking at the reported standard deviation of the results in XMem [21]. Similarly, in this thesis, the means and standard deviations of six runs of the proposed methods in Chapters 4, 5, and 6 are reported in Table 7.1.

In [21], authors compare the performance of different methods by taking the average of five runs; however, they did not report the five update steps that they used. Comparing the standard deviations of JOINT in Table 7.1 with those reported in [21], we can see that the selected six memory update steps are reasonably close to those in [21].

As seen in Table 7.1, the proposed solutions improve the performance of both selected Online VOS baseline models (LWL and JOINT) on long videos when the objects in the video have a long trajectory with sudden representation drifts. Furthermore, as illustrated in Table 7.1, by comparing LWL and LWL-GRCL methods, we can deduce that the proposed prior-focused GRCL can improve the robustness of the LWL model against selecting different values for target model update size Δ_{C}, evident by the lower reported standard deviation in table; however, we will see in Section 7.5.1 that is not generally true.

It is worth mentioning that JOINT has a parallel transduction branch in its structure, which benefits from a transformer model that acts like a matching-based method. This

is an important factor leading the proposed GRCL to be less effective in reducing the standard deviation of JOINT-GRCL performance. Although the transduction branch of JOINT can boost the positive or even negative effects of the proposed solutions, the average performance $\mathcal{J}\&\mathcal{F}$ of JOINT-GRCL is improved significantly by 3% relative to the JOINT baseline.

For LWL, RMSCL improves the robustness of the model against different target model update step size Δ_C by decreasing the standard deviation of LWL in LWL-RMSCL. The effect of RMSCL on JOINT is on both branches of JOINT's structure, which amplifies the positive and negative effects of JOINT and consequently increases the standard deviation of JOINT performance; however, it outperforms JOINT by almost 8% on the Long Videos dataset. On LWL, RMSCL improves the accuracy of the baseline by more than 3%.

The performances of Hybrid methods on baselines are also shown in Table 7.1. The proposed Hybrid solution improves the robustness (smaller standard deviation) of LWL with the most significant accuracy improvement of more than 6%. JOINT-Hybrid also outperforms both the accuracy and robustness of JOINT-RMSCL and JOINT-GRCL. We will discuss the performance of the Hybrid approach on Figures 7.2 and 7.3.

To have a fair comparison, the proposed methods and the baseline Online VOS frameworks are compared with four matching-based methods, including RMNet [72], STM [20], STCN [73], and the current VOS state-of-the-art approach XMem [21]. The reported results of the matching-based methods are from [21]. STM is a matching-based VOS baseline that has been a state-of-the-art method for a long period of time in VOS and RMNet, STCN and XMem are its follow-up methods. RMNet and STCN try to improve the memory functionality of STM by having better memory encoding and memory reading methods. XMem also be considered an extension of STM which is specifically designed to work on long video sequences.

Figures 7.2 and 7.3 demonstrate the comparison between the performances of each 6 runs based on different memory and target model update step sizes while the two update step sizes are the same for this experiment ($\Delta_C = \Delta_{\mathcal{M}}$). Figures 7.2 and 7.3 shows first eight methods' performance of Table 7.1 on LWL and JOINT respectively. In addition to those eight performances, LWL-No-Update and JOINT-No-Update show the performance of the two baselines method without updating the target model C on the extracted information of the evaluated frames. These two results show the impact of the given ground truth label alone on the Long Videos dataset.

As shown in Figures 7.2, LWL-GRCL outperforms LWL with a large margin when the memory and target model step size are small ($\Delta_C = \Delta_{\mathcal{M}} = \{1, 2, 4, 6\}$) whereas, for a bigger target model and memory update step size ($\Delta_C = \Delta_{\mathcal{M}} = \{8, 10\}$), the performance

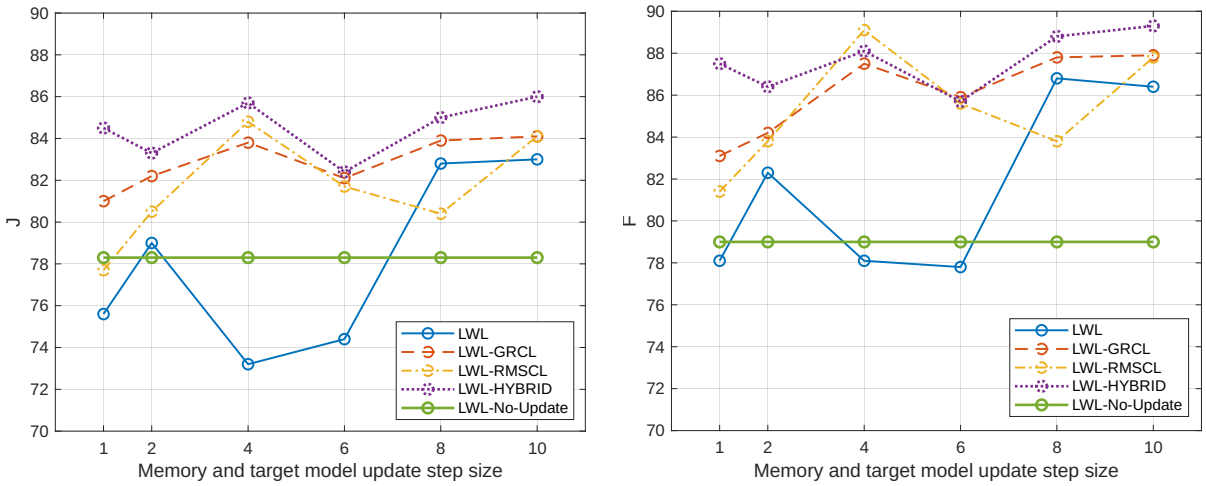

Figure 7.2: Performance comparison of competing methods as a function of memory and target model update step sizes, ($\Delta_{\mathrm{C}} = \Delta_{\mathcal{M}}$), on the Long Videos dataset [1]. The left figure shows $\mathcal{J}$ and the right one illustrates $\mathcal{F}$ over 6 runs for LWL and LWL augmented by three proposed methods. The green line shows the performance of LWL without updating its target model on the memory.

of LWL-GRCL is better than LWL with a smaller margin. This is because of having a more diverse memory $\mathcal{M}$ with a bigger memory step size $\Delta_{\mathcal{M}}$ that makes the updating target model alone without GRCL regularization effective.

In Figures 7.2 and 7.3, the impact of the Hybrid solution is more highlighted than the results in Table 7.1, as LWL-Hybrid and JOINT-Hybrid have better robustness against different update step sizes. As you can see in Figure 7.3, the Hybrid methods have fewer fluctuations in performance in comparison to the prior and likelihood-focused methods (JOINT-GRCL and JOINT-RMSCL). Additionally, the Hybrid approach is the best approach among the proposed continual learning approaches in this thesis in terms of performance ($\mathcal{J}$ and $\mathcal{F}$) on both baseline methods (LWL and JOINT).

It is worth nothing that with a small target model update size, the Hybrid methods could implicitly benefit from their GRCL part, and with a large target model update size ($\Delta_{\mathrm{C}} = \{6, 8, 10\}$), the likelihood-focused part (RMSCL) of the Hybrid approach mostly improves the performance of the baseline. In better words, the proposed posterior-focused solution (Hybrid) counts on its likelihood when there is enough data available in the memory, and it counts on its prior-focused part (GRCL) when the memory does not contain

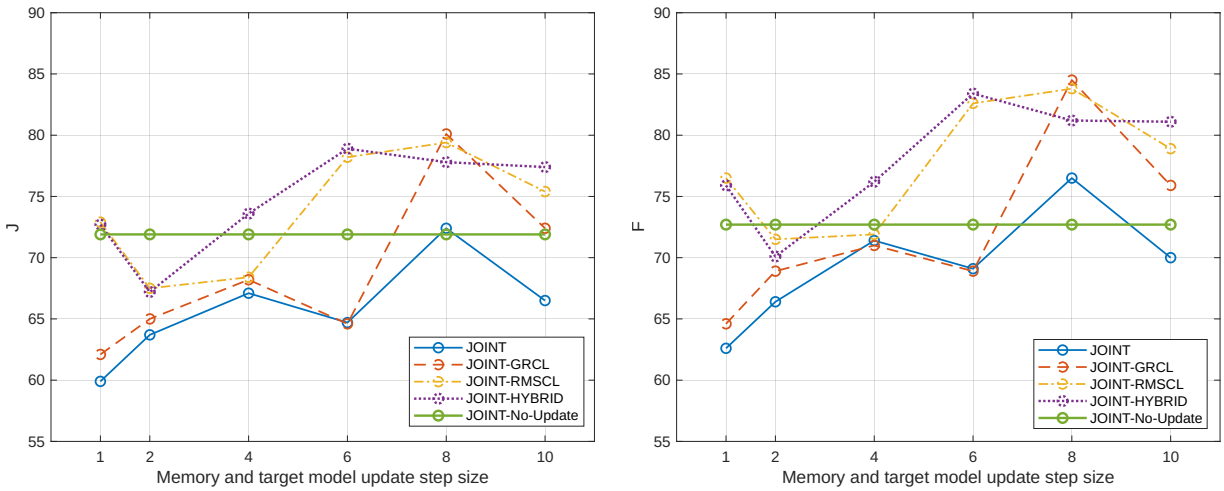

Figure 7.3: Performance comparison of competing methods as a function of memory and target model update step sizes on the Long Videos dataset [1]. Each figure illustrates adopted JOINT's performance ($\mathcal{J}$ and $\mathcal{F}$) over 6 runs. The green line shows the performance of JOINT without updating its target model on the memory.

enough data from the past. Note that, by "enough", I mean enough diverse data samples that are a good representative of the video sequence that its frames are evaluating.

RMSCL has a more positive effect on JOINT since it provides the updated working memory $\mathcal{M}_W$ for its transduction branch as well. It is interesting to see that LWL outperforms JOINT on the Long Videos dataset, and the reason could be the lack of generalization in its offline-trained parts, specifically its transformer part.

Figure 7.4 shows the qualitative results of the proposed methods (GRCL, RMSCL, and Hybrid) and baselines (LWL and JOINT) on 7 selected frames of the "dressage" video sequence from the Long Videos dataset. The results in Figure 7.4 are produced by applying the evaluated methods to the Long Videos dataset when $\Delta_{\mathrm{C}} = \Delta_{\mathcal{M}} = 1$. The results show RMSCL improves the performance of LWL and JOINT; however, GRCL improves the performance of LWL but cannot improve the performance of JOINT on the selected frames. Thus, GRCL improves the performance of the baselines if the prior information is correct (it happens with LWL). As shown in the figure, LWL-Hybrid has the best performance on the Long Videos dataset; however, baseline methods are more vulnerable to the distribution drift of the target object. The distribution drifts that happen on the "dressage" video are shown in Figure 2.4 in Chapter 2.

Figure 7.4: Qualitative comparison of the competing frameworks in the context of the Long Videos dataset [1]. The associated frame number for each image is shown at the bottom of the figure. The leftmost column shows the given mask Y_g, which is the same for all of the methods. The results show that the proposed methods, when augmenting the baseline frameworks, can lead to robust representation drift. The missed part of the estimated object mask is shown in gray while the segmented mask is shown in pink. Additionally, the frameworks based on RMSCL (LWL-RMSCL, JOINT-RMCSL) are less vulnerable to the distribution changes that take place in long video sequences. Finally, as shown in the figure, LWL-Hybrid has the best performance among all the proposed methods.

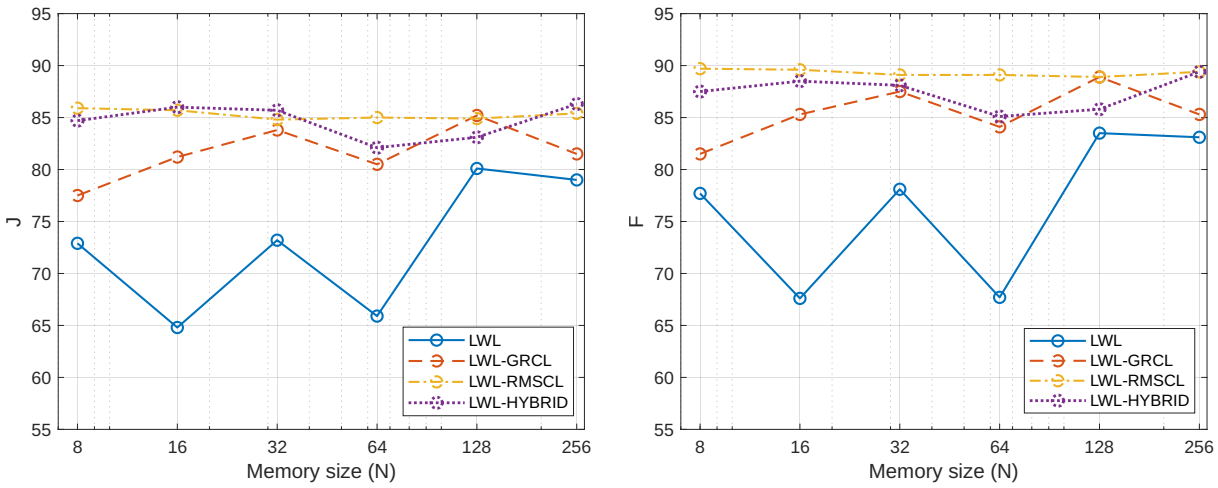

Figure 7.5: The effect of different memory size N on the proposed methods compared to the baseline (LWL) on the Long Videos dataset [1]. The performance of the LWL baseline fluctuates with memory size (N), while LWL-GRCL, LWL-RMSCL, and LWL-RMSCL are more robust to the selection of different memory size N. Here, the target model and memory update step are $\Delta_{\mathcal{M}} = \Delta_{\mathrm{C}} = 4$.

On long video sequences, it is not feasible to store all of the previously evaluated frames' information in the memory $\mathcal{M}$, as such, it is important to limit the memory size N. Here, we aim to evaluate how different memory sizes affect baselines and the proposed methods. For this experiment, we compare the performance of LWL, LWL-RMSCL, LWL-GRCL, and LWL-Hybrid on the Long Videos dataset $N \in \{8, 16, 32, 64, 128\}$ and the target model and memory update step are $\Delta_{\mathcal{M}} = \Delta_{\mathrm{C}} = 4$. As seen in Figure 7.5, increasing the memory size N improves the performance of the methods, but it also increases the computational complexity of the evaluated Online VOS methods. Increasing the size of memory $\mathcal{M}$ does not have a considerable effect on LWL-RMSCL since it does not have any hyper-parameters that are affected by the size of $\mathcal{M}$; however, tuning the hyper-parameters (thresholds) of LWL-GRCL and LWL-Hybrid is implicitly affected by the size of the memory N. This is the reason that LWL-GRCL and LWL-Hybrid performance fluctuated with changing the size of $\mathcal{M}$. RMSCL on the other hand, provides a small set of diverse enough data with new weights Ψ in its dynamic working memory $\mathcal{M}_W$ which improves both accuracy and speed of the baseline methods on long video datasets.

Figure 7.6 illustrates how increasing the memory size N affects the speed, measured

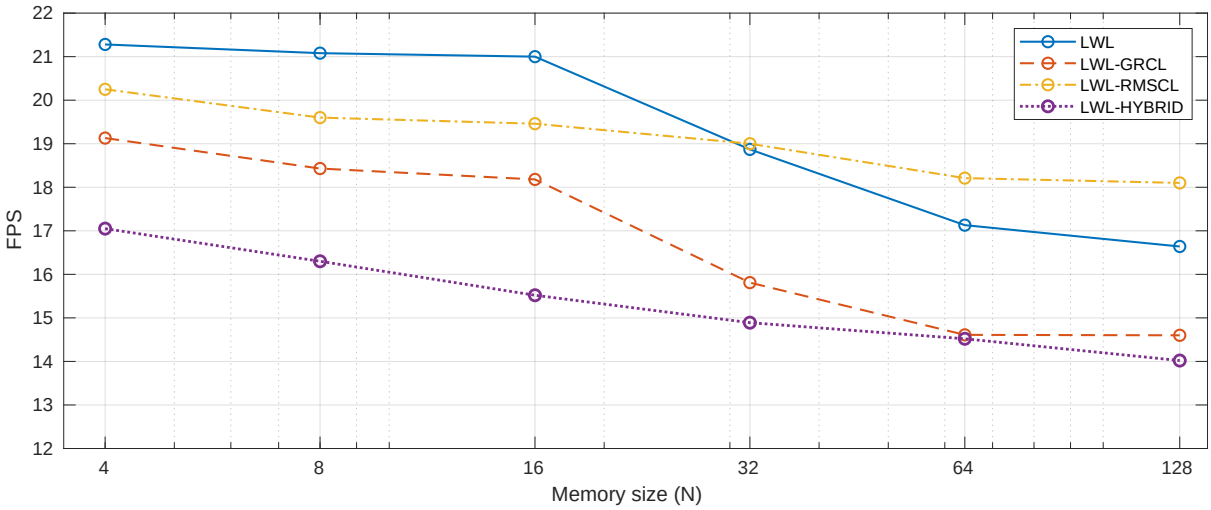

Figure 7.6: Run-time evaluation: The proposed methods' run-times are compared against the LWL baseline on the DAVIS16 dataset. LWL-RMSCL leads to higher Frame Per Second (FPS) when the memory size N is increased. The memory and the target model update step is set to $\Delta_{\mathcal{M}} = \Delta_{\mathrm{C}} = 1$ for all of selected N.

in FPS, of the evaluated methods (LWL, LWL-GRCL, LWL-RMSCL, and LWL-Hybrid) on DAVIS16. The memory and the target model update step is set to $\Delta_{\mathcal{M}} = \Delta_{\mathrm{C}} = 1$ for the results in Figure 7.6. As shown, the FPS of LWL-RMSCL is degraded less than LWL, LWL-GRCL and LWL-Hybrid while LWL-GRCL and LWL reported almost the same degradation with increasing the memory size N. Since LWL-RMSCL uses smaller working memory $\mathcal{M}_W$ for training the target model, it would be faster than LWL and LWL-GRCL when the memory size is increased (bigger than 32). It is worth mentioning that minimizing Equation (5.3) in the RMSCL approach is affected by increasing the memory size N and consequently it affects the FPS of LWL-RMSCL as well. LWL-Hybrid has the lowest FPS among the proposed approaches since it has the computational complexity of both GRCL and RMSCL. When the memory size (N) is increased, the Hybrid approach degrades at the same rate as RMSCL.

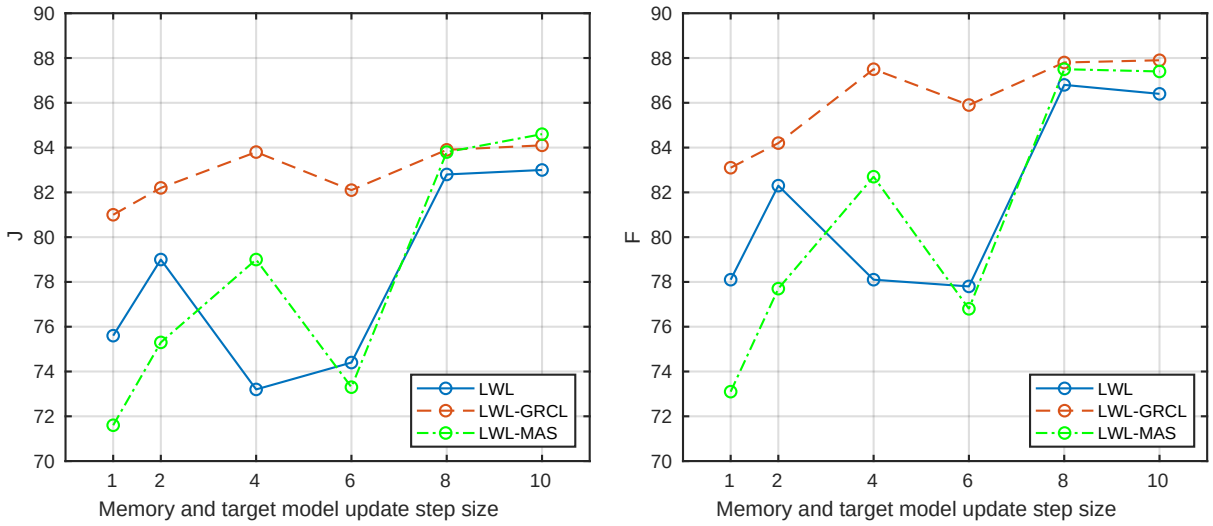

Figure 7.7: Quantitative evaluation ($\mathcal{J}$ left, $\mathcal{F}$ right) of the proposed GRCL with a conventional continual learning (MAS) [49] on the Long Videos dataset [1]. As seen, standard MAS [49] is not as effective as the proposed GRCL when incorporated into the Online VOS framework.

7.4.2 Conventional Continual Learning

One important aspect of the proposed prior-focused continual learning method to augment Online VOS frameworks is that it be customized and designed specially for Online VOS. To illustrate that, the performance of the proposed method (LWL-GRCL) is compared against the LWL framework when is augmented with standard MAS continual learning regularizer [49] as a prior-focused regularizer for updating the target model (LWL-MAS). The evaluation is conducted on the Long Videos dataset, where the results are demonstrated in Figure 7.7. As shown in Figure 7.7, LWL-GRCL reports higher average performance $\mathcal{J}$ and $\mathcal{F}$ compared to LWL-MAS. Two main reasons can be provided to further elaborate the reported gap on the performance of these two compared frameworks: i) The overall gated-regularized map $\mathbf{G}^{t-1}$ described in Figure 5.1 and Algorithm 3 keeps the efficiency of the proposed GRCL compared to the MAS approach, where the MAS regularizer loses its efficiency when the update steps are increased, as explained in Section 4.1. MAS highly benefits from Ω^t, however the efficiency of Ω^t is being degraded as more and more target model gradients are processed and stored over time. It causes all of the parameters to become important as the number of updates increases. On the other hand, LWL-GRCL with

its dynamic memory size, guarantees that the target model C has enough free parameters to learn new tasks. ii) For a small number of training epochs, in each updating step of C^t the binarized regularizer $\mathbf{G}^{t-1}$ (hard regularizer) is more effective than MAS with a soft regularizer Ω^t.

7.4.3 Short Video Evaluation

Table 7.2 demonstrates the performance of the adopted Online VOS frameworks based on the proposed approaches and competing methods on short video datasets (i.e., DAVIS16, DAVIS17, and YouTube-VOS18). The same hyper-parameters are used for short and long videos, meaning that the models do not have prior knowledge of the length of the video sequence being processed. As mentioned in Section 2.4.2, objects in short video datasets have a short trajectory and their representations are mostly kept intact or gradually changing through the frames. As seen in Table 7.2, the augmented frameworks by the proposed prior-focused GRCL perform the same as the baseline methods, and the proposed regularizer not only does not affect the performance of the baseline method when there is no representation drift on objects in videos, but also LWL-GRCL performs slightly better compared to LWL on YouTube-VOS18.

Table 7.2: Performance analysis of the evaluated methods against the validation sets of DAVIS16, DAVIS17, and YT-VOS18.

Method	YT-VOS 2018					DAVIS17			DAVIS16			FPS
	$\mathcal{J}_s$	$\mathcal{F}_s$	$\mathcal{J}_u$	$\mathcal{F}_u$	$\mathcal{G}$	$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}\&\mathcal{F}$	
LWL [16]	79.5	83.9	75.7	83.4	80.6	77.1	82.9	80.0	87.3	88.5	87.9	18.87
LWL-GRCL (ours)	79.7	84.0	75.9	83.9	80.9	77.0	83.0	80.0	87.3	88.6	88.0	15.89
LWL-RMSCL (ours)	79.4	84.0	75.2	83.2	80.5	75.6	81.8	78.7	86.5	88.3	87.4	19.01
LWL-Hybrid (ours)	78.8	83.2	74.5	82.6	79.8	75.5	81.7	78.7	86.5	88.4	87.5	14.91
JOINT [17]	81.6	85.6	78.6	86.0	82.9	80.6	86.0	83.3	87.5	89.4	88.5	6.21
JOINT-GRCL (ours)	81.6	85.5	77.6	84.9	82.4	80.4	85.8	83.2	87.5	89.4	88.5	6.09
JOINT-RMSCL (ours)	81.0	85.0	77.6	84.9	82.2	79.8	85.4	82.6	87.9	90.0	89.0	11.15
JOINT-Hybrid (ours)	81.0	85.1	77.3	84.8	82.1	79.9	85.4	82.6	87.8	89.9	88.9	9.06
RMNet [72]	82.1	85.7	75.7	82.4	81.5	81.0	86.0	83.5	88.9	88.7	88.8	11.9
STM [20]	79.7	84.2	72.8	80.9	79.4	79.2	84.3	81.8	88.7	89.9	89.3	14.0
STCN [73]	81.8	86.5	77.9	85.7	83.0	82.2	88.6	85.4	90.8	92.5	91.6	26.9
XMem [21]	84.6	89.3	80.2	88.7	85.7	82.9	89.5	86.2	90.4	92.7	91.5	29.6

In Table 7.2, the baseline models' parameters suggested in their respective original publications are utilized for reporting $\mathcal{J}$, $\mathcal{F}$ and FPS. On short videos, one target model update step is selected for each dataset, in contrast to the results on the Long Videos dataset, where a set of 6 target model update step sizes are used. For JOINT, C is updated every 3 frames ($\Delta_{\mathrm{C}} = 3$), and for LWL, C is updated every frame ($\Delta_{\mathrm{C}} = 1$); however, XMem updates its so-called working memory every 5 frames ($\Delta_{\mathcal{M}} = 5$). All of the mentioned parameters are suggested for their best performances on short videos suggested in their papers [16, 17]. The proposed RMSCL slightly degrades the performance of LWL on all of the short video datasets, but it improves the speed of the baselines, specifically JOINT because both its branches are affected by RMSCL. In JOINT-RMSCL both its online learning part and its transformer part use $\mathcal{M}_W$ and that is why JOINT-RMSCL reports more improvement in speed (higher FPS) in comparison with JOINT. Table 7.2 also shows the baselines perform slightly better in terms of FPS since GRCL needs to calculate a new G^t after every updating step t; however, for a small target model C^t this FPS degradation is not significant.

The Hybrid method also gets almost the same performance as RMSCL on short videos; however, the Hybrid method is slower than RMSCL since it has the computational complexity of both GRCL and RMSCL.

The performance degradation of the proposed methods on short video datasets in all cases except LWL-RMSCL and LWL-Hybrid are negligable (less than 1%); however, the impacts of the proposed methods on the Long Videos dataset make it reasonable to use them on the Online VOS methods. LWL-RMSCL and LWL-Hybrid also decrease the performances of LWL on the short video datasets less than 2% .

Figure 7.8 demonstrates the qualitative results of the proposed methods and baselines on the the "soapbox" video sequence of DAVIS16 [86]. As illustrated, the proposed continual learning methods offer positive improvement on JOINT shown with red border in Figure 7.8 with slight and tiny changes in LWL results, which is in agreement with the reported results in Table 7.2. The "Soapbox" video is one of the longest video sequences of DAVIS16 with 99 frames.

7.4.4 Memory Efficiency

To compare the memory efficiency of the proposed prior-focused GRCL against the baseline, we compare each unit of memory $\mathcal{M}$ of LWL and each memory unit of $\mathcal{M}$ of adopted LWL-GRCL. In LWL, each sample in the memory $\mathcal{M}$ consists of the preceding estimated object masks $\mathcal{Y}$ and its related input frames' extracted features $\mathcal{X}$. Each feature $X \in \mathcal{X}$

Figure 7.8: The qualitative comparison of the evaluated methods on a short video dataset (DAVIS16 [86]). The results show that the proposed GRCL, RMSCL, and Hybrid almost keep the performances of baselines intact on DAVIS16 and improve the object segmentation of some frames shown in red compared to JOINT. These qualitative results reflect the quantitative results of Table 7.2.

has a dimension of $512 \times 30 \times 52$ floats (64 bits). In contrast, each binary regularized-gated map (G) has a dimension of $512 \times 16 \times 3 \times 3$ bits. Moreover, each unit of $\mathcal{M}$ also has a binary mask of the target model C output size of 30×52. As a result, each unit of $\mathcal{M}_G$ is almost 693 times smaller than each unit of $\mathcal{M}$. This comparison would be more important if there was a need to improve the performance of an Online VOS implemented on a small device. Thus, having a larger gated-regularizer memory $\mathcal{M}_G$ is less expensive than having a large memory $\mathcal{M}$.

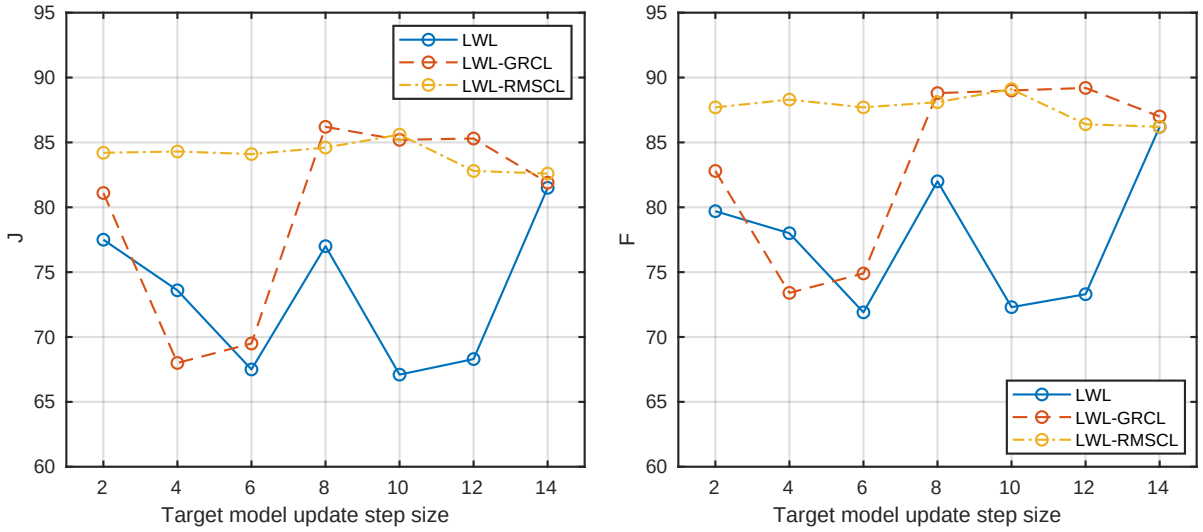

Figure 7.9: The effect ($\mathcal{J}$ left, $\mathcal{F}$ right) of target model update step size Δ_{C}: The competing methods are evaluated via Long Videos dataset [1]. The results show that the proposed LWL-GRCL and LWL-RMSCL are more robust to different target model update step sizes Δ_{C} when the memory update step size is fixed $\Delta_{\mathcal{M}} = 1$.

7.5 Ablation study

In this section, the effects of some key parameters of the proposed methods are evaluated on the adopted method's performance. Additionally, the effects of some part of the proposed methods also will be discussed in this section. The comparisons are mainly on the LWL approach when augmented with prior-focused GRCL, likelihood-focused RMSCL, and Hybrid. This section shows a range of experimental results and ablation studies on the Long Videos dataset.

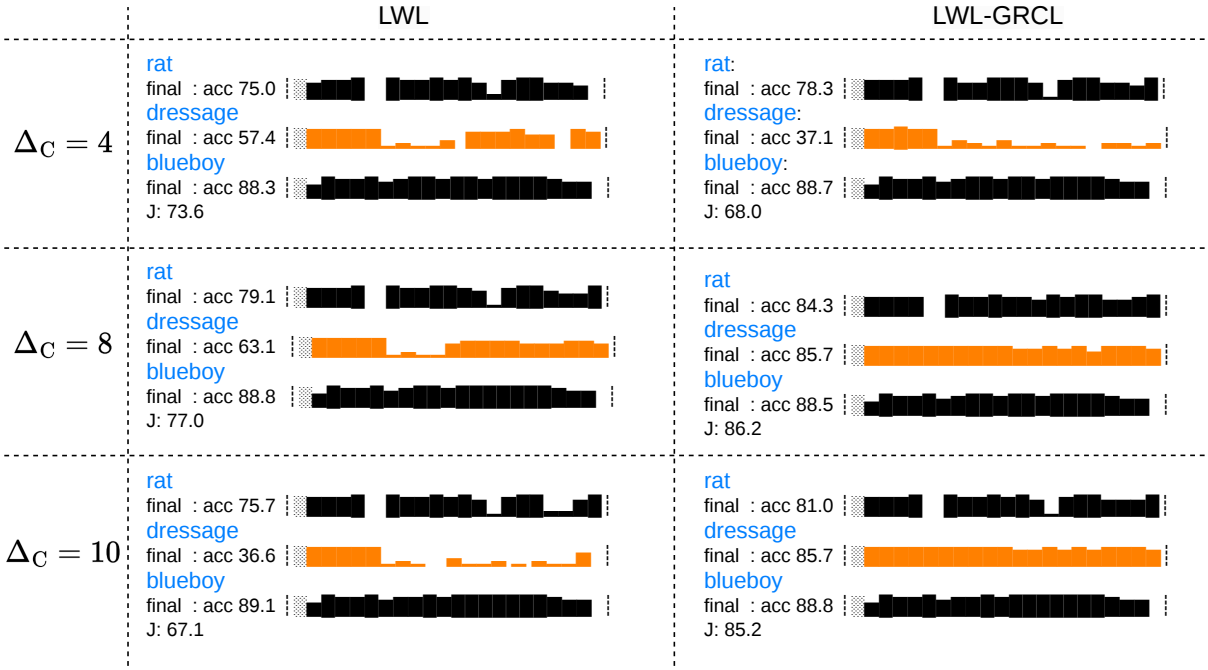

Figure 7.10: Per-class and per-frame performances ($\mathcal{J}$) of LWL and LWL-GRCL on the Long Videos dataset [1] when $N = 4$, $\Delta_C \in \{4, 8, 10\}$, and $\Delta_\mathcal{M} = 1$. The figure shows the segmentation scores of each method on the labelled frames of three videos of the Long Videos dataset. This figure shows that LWL-GRCL has difficulties recovering from incorrect segmentation in the "dressage" video ($\Delta_C = 4$), whereas it helps LWL avoid this situation ($\Delta_C \in \{8, 10\}$).

7.5.1 Target Model Update Step Size Δ_C

To justify the effect of target model update step size on the proposed methods, an ablation study is conducted to compare the performance of LWL-GRCL, LWL-RMSCL and LWL on the Long Videos dataset. Here, the memory update step size is fixed to $\Delta_\mathcal{M} = 1$ and only the target model update step size varies $\Delta_C \in \{2, 4, 6, 8, 10, 12, 14\}$. It is worth noting

that, in all of the results in Section 7.4, memory $\mathcal{M}$ and C^t were updated sequentially but at the same time index ($\Delta_{\mathrm{C}} = \Delta_{\mathcal{M}}$). For this experiment, the memory size is set to $N = 4$ to make the situation hard for all of the evaluated methods. As seen in Figure 7.9, LWL's performance has the lowest performance in comparison to LWL-GRCL and LWL-RMSCL. LWL also experiences a drop in the update step size $\Delta_{\mathrm{C}} \in \{6, 10, 12\}$, but by undertaking the proposed methods, the degree of this degradation of performance is decreased, except for LWL-GRCL at $\Delta_{\mathrm{C}} = 4$. This degradation of performance by GRCL shows a drawback of the prior-focused method when the model focuses on a wrong prior and that wrong prior is kept and intensified through the evaluation of future frames.

The first row of Figure 7.10 shows the details of the performance degradation of LWL-GRCL in Figure 7.9 when $\Delta_{\mathrm{C}} = 4$. The per-calss and per-frame performances of LWL and LWL-GRCL are shown as a bar plot of the segmentation scores $\mathcal{J}$ and compared in Figure 7.10. The figure shows LWL-GRCL cannot recover from a wrong segmentation because of the blind parameter regularization of GRCL; however, GRCL improves the performance of LWL on the other two target model step sizes ($\Delta_{\mathrm{C}} \in \{8, 10\}$). In other words, GRCL helps Online VOS not to do wrong object segmentation, but if it does, and usually with small target model update sizes, the important parameters of the target model being trained on the wrong object segmentation frames would be kept intact and remembered along the video frames.

It is worth noting that for this experiment, the hyper-parameters of GRCL are the same as the hyper-parameters that are tuned for LWL setup with memory $N = 32$. As shown in Figure 7.5, LWL has more difficulty segmenting the object in long video sequences with a considerably limited memory size.

In Figure 7.9, the impact of the proposed RMSCL method is more considerable. RMSCL has lower performance in comparison to GRCL with large step sizes $\Delta_{\mathrm{C}} \in \{8, 10, 12, 14\}$) since the memory is not diverse enough and RMSCL cannot provide any better compact version of a working memory $\mathcal{M}_W$ for the baseline. It is worth mentioning that the impact of RMSCL with a small memory size $N = 4$ is more initiated by its re-weighting mechanism of using Ψ^t instead of d_n as explained in Section 5.2. Next, the ablation studies related to GRCL are discussed.

7.5.2 GRCL

For the experimental results in Section 7.4, the gated-regularizer memory size P is varying during the evaluation time, which makes the gated-regularizer memory $\mathcal{M}_G$ dynamic. In

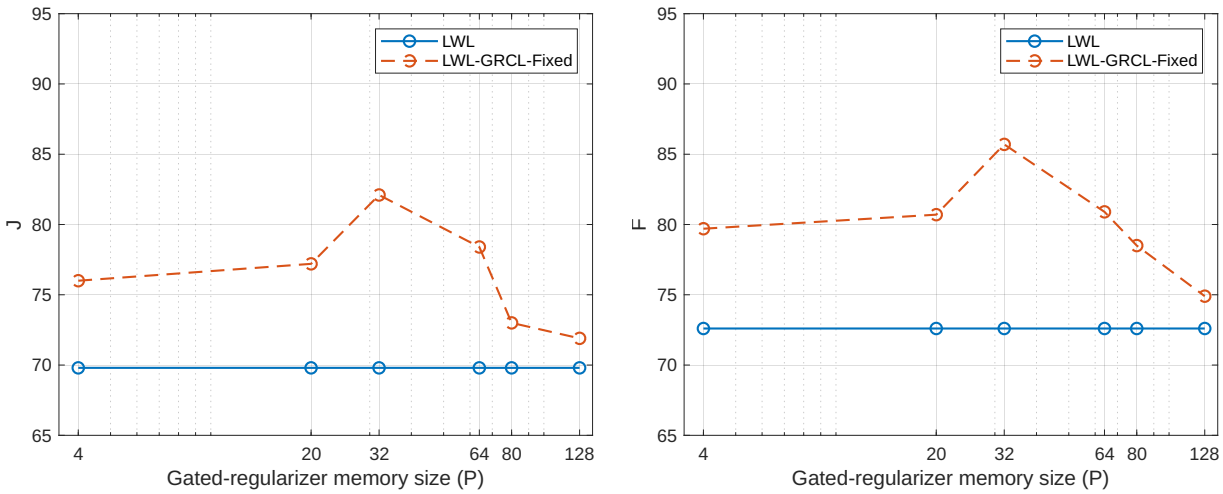

Figure 7.11: The effect of gated-regularizer memory size P on the LWL-GRCL-Fixed framework. For this experiment, the memory size is fixed and limited to $(N = 8)$ in order to properly analyze the impact of the proposed GRCL-Fixed. The experiment also is done on Long Videos dataset. By setting P to a large number, the target model C^t will not have enough free parameters to be updated on memory $\mathcal{M}$.

GRCL, the size of $\mathcal{M}_G$ depends on the regularized parameters of the target model C^t, if it crosses two certain thresholds, the gated-regularizer memory size will be adjusted.

First, in order to show that $\mathcal{M}_G$ should be dynamic, an experiment is done considering a static gated-regularizer memory $\mathcal{M}_G$ size and with this change the prior-focused approach is named GRCL-Fixed. Next, the effect of $\mathcal{M}_G$ size is discussed on LWL-GRCL-Fixed.

Gated-Regularizer Memory Size

Figure 7.11 shows the performance of LWL-GRCL-Fixed with different gated-memory sizes $P \in \{4, 20, 32, 64, 80, 128\}$ on the Long Videos dataset. As demonstrated in Figure 7.11, increasing P improves the performance of LWL-GRCL until the number of regularized parameters does not degrade target model learning. For the main experimental results in Section 7.4, the gated-regularizer memory size is varying based on the number of regularized parameters of the target model, and it makes $\mathcal{M}_G$ dynamic in size where the memory $\mathcal{M}$ size is fixed and set to $N = 32$. The reason for limiting the memory size to $N = 8$ is to highlight the impact of gated-regularizer memory. In other words, we need to limit

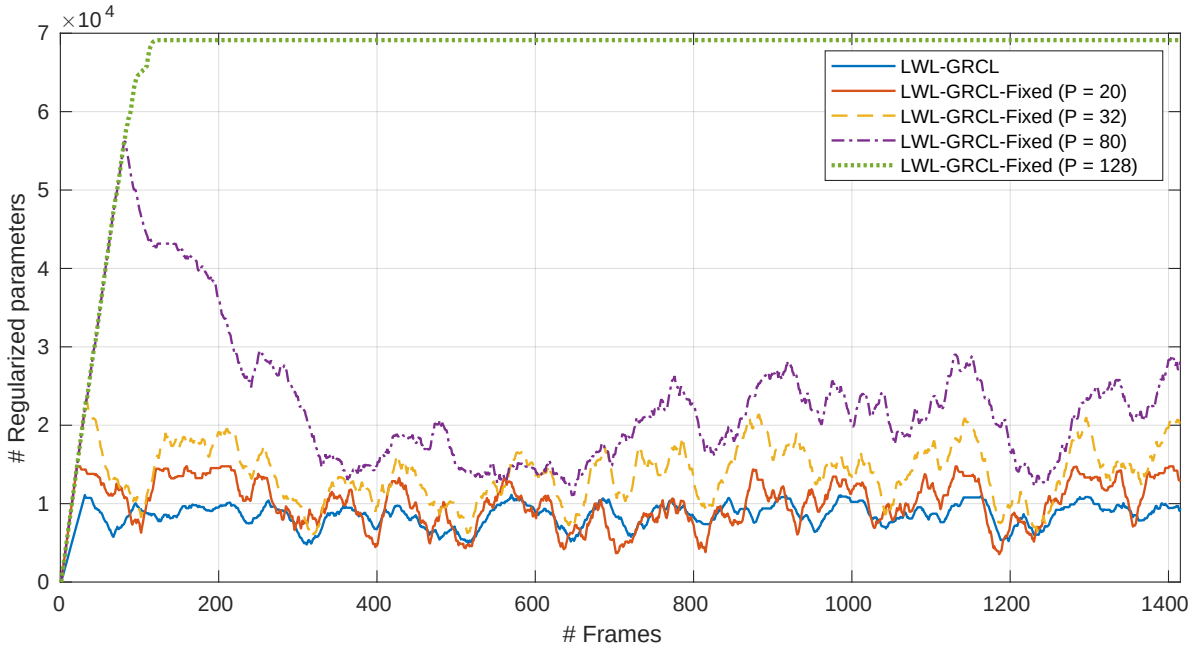

Figure 7.12: The number of regularized target model parameters Θ when incorporated into LWL-GRCL-Fixed, as a function of gated-regularizer memory size ($P = \{20, 32, 80, 128\}$). The number of regularized parameters of LWL-GRCL are shown with solid blue line. The results are based on 1416 frames of the "rat" video sequence of the Long Videos dataset [1]. For this experiment, C is updated every frame ($\Delta_{\mathrm{C}} = 1$), and the memory size is set to ($N = 8$).

the memory size N to make more severe forgetting happens on the video sequence. This effect can also be seen from another perspective in Figures 7.2 where, with a small memory update step size ($\Delta_{\mathrm{M}} < 4$), LWL-GRCL improves the baseline considerably, assuming with a small memory update step size the memory is updated with similar information more frequently.

Regularized Parameters of the Target Model

The number of regularized parameters in C^t is important factor related to the ability of the target model to learn new information. As seen in Figure 7.12, the regularized parameters of the target model C^t is increased, while the gated-regularizer memory $\mathcal{M}_G$ is growing by evaluating new frames (the evaluation of first P frames). This growth in

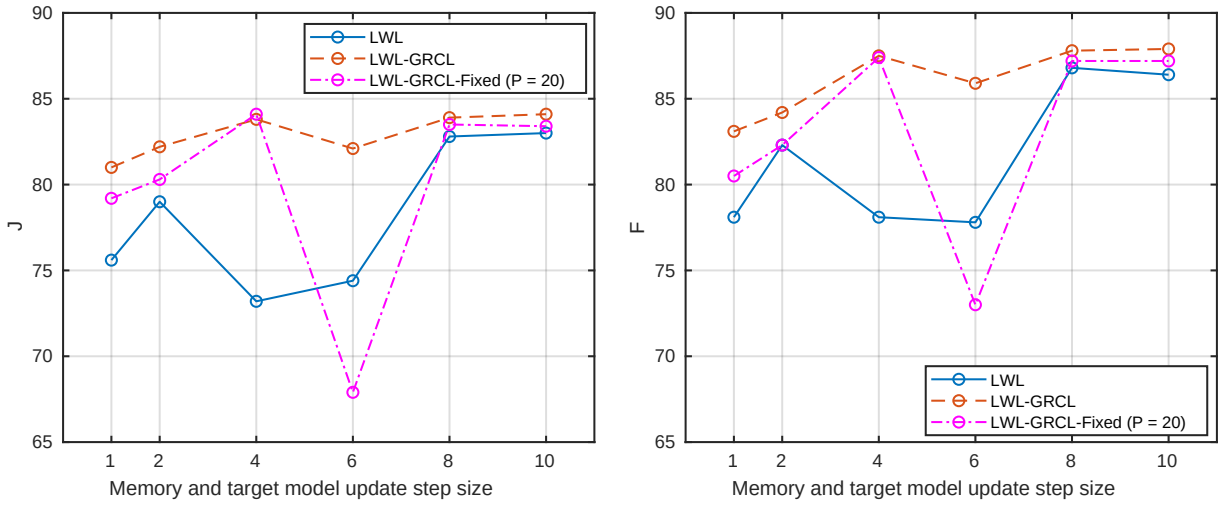

Figure 7.13: Quantitative evaluation ($\mathcal{J}$ on the left and $\mathcal{F}$ on the right) of the proposed GRCL with GRCL-Fixed (p=20) on the Long Videos dataset [1]. As seen, GRCL with dynamic gated-regularizer memory $\mathcal{M}_G$ has better performance than GRCL-Fixed with a constant size $\mathcal{M}_G$.

the number of regularized parameters is different for different P for GRCL-Fixed where the gated-regularizer memory size is fixed. For $P = 128$, almost all of the parameters of C are regularized, and in this case C^t does not have any free parameters to be trained, and even removing or replacing one gated-regularization map G^j from $\mathcal{M}_G$ would not free enough parameters and solve the problem. In other words, C^t would not have enough free parameters to be updated on the new updated memory $\mathcal{M}^t$. When the gated-regularizer memory $\mathcal{M}_G$ reaches its maximum capacity, the oldest G in the gated-regularizer memory will be replaced by the next gated-regularizer map G^t. This will free up some parameters since G^t has a lower number of "1" in its map in comparison with the oldest G in the memory. This decreasing number of regularized parameters in the overall gated-regularizer $\mathbf{G}^{t-1}$ will continue untill it reaches the balance number, and then this phenomenon will continue in a periodic order. This can be seen in Figure 7.12 on LWL-GRCL-Fixed (P=80). The same pattern with smaller intensity happens to the other two, LWL-GRCL-Fixed (P=32) and LWL-GRCL-Fixed (P=20). To address the discussed issue in GRCL-Fixed, a mechanism is proposed for GRCL that makes $\mathcal{M}_G$ dynamic in size. For LWL-GRCL, no P is set, and as shown in Figure 7.12 and as you can see, the number of the target model's regularized parameters is bounded between 5000 and 11000. These two numbers

are set based on two GRCL hyper-parameters (ξ_l and ξ_u) which are set to 0.07 and 0.15, respectively. The results in this section show it is more meaningful to provide a gated-regularizer memory $\mathcal{M}_G$ with dynamic size for GRCL.

Dynamic Gated-Regularizer Memory

Based on the discussions about the impact of $\mathcal{M}_G$ size and its relationship to the number of regularized parameters of the target model, GRCL benefits from a dynamic $\mathcal{M}_G$ size using two thresholds ξ_l and ξ_u which were discussed in the Section 4.2.1. Figure 7.13 shows the comparison between LWL-GRCL with dynamic $\mathcal{M}_G$, LWL-GRCL-Fixed with a static $\mathcal{M}_G$ with size $P = 20$, and LWL. For LWL-GRCL-Fixed, the size of gated-regularizer memory is set to $P = 20$ as the closest P which regularizes almost the same number of parameters as GRCL as shown in Figure 7.12. The result shows the superiority of LWL-GRCL over other methods on the Long Videos dataset.

Figure 7.13 again illustrates the main important limitation of the prior-focused method when the prior information is not correct. As shown in Figure 7.10, this problem happens when an update is done on the wrong segmented targets, and GRCL wants to remember that update for segmenting future frames of video. Clearly, it happens on LWL-GRCL-Fixed (P=20) when $\Delta_{\mathcal{M}} = \Delta_{\mathrm{C}} = 6$; however, for LWL-GRCL with dynamic $\mathcal{M}_G$ it does not happen. It is worth noting that the chosen hyper-parameters of GRCL are selected empirically by conducting multiple tests with the goal of increasing the performance of selected Online VOS methods (LWL and JOINT) on the Long Videos dataset and maintaining the performance of the model on short video datasets.

7.5.3 RMSCL

Working Memory Size

The reconstruction-based memory selection part of RMSCL selects samples from memory $\mathcal{M}^t$ and places them in the working memory $\mathcal{M}^t_W$ for updating the target model C^{t-1} to form the target model C^t. This memory selection mechanism provides a smaller an more diverse working memory in comparison to Online VOS that uses $\mathcal{M}^t$ for the target model C^t. The performance of RMSCL and the Hybrid method is strongly dependent on the quality of the selected samples in the working memory. Figure 7.14 shows the number of selected samples from $\mathcal{M}$ in every update step of LWL-RMSCL on the "rat" video of the Long Videos dataset. In this figure, the target model and the memory $\mathcal{M}$ is updated every frame ($\Delta_{\mathcal{M}} = \Delta_{\mathrm{C}} = 1$).

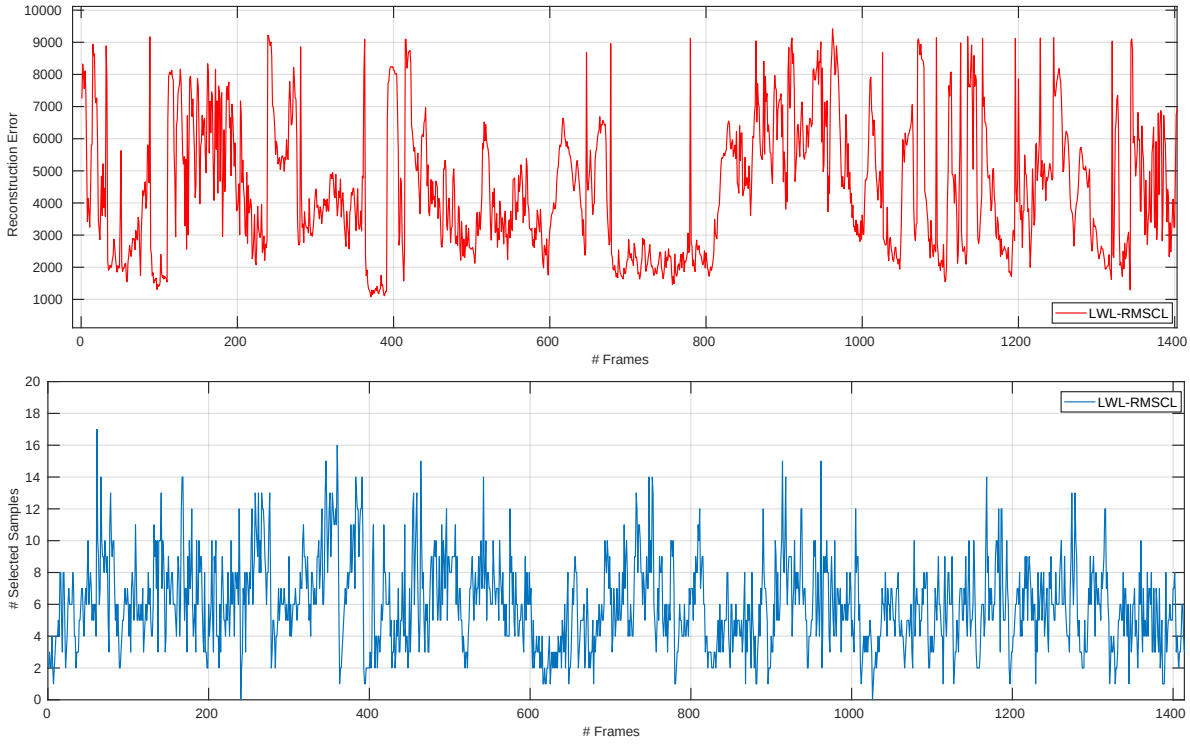

Figure 7.14: The reconstruction error and the number of selected samples for each update step in the RMSCL method on the "rat" video of the Long Videos dataset [1]. As seen in bottom plot of this figure, LWL-RMSCL usually select small number of selected samples with the mean of 6 from the memory of size 32 which decreases the computational complexity of LWL-RMSCL in comparison to LWL that updates the target model C on 32 samples of $\mathcal{M}$.

As illustrated in Figure 7.14, the required samples for updating the target model is less than all of the available data in the memory $\mathcal{M}$ ($N = 32$). It is worth noting that in RMSCL, the given frame feature and segmented labels always are placed in the memory with a fixed weight. Additionally, the reconstructed error that is shown in Figure 7.14 indicates that the Reconstruction errors of the current frame's feature X_{t+1} is usually between 2000 to 9000. This results justify the message-passing mechanism of the Hybrid approach and will be discussed more in the next section.

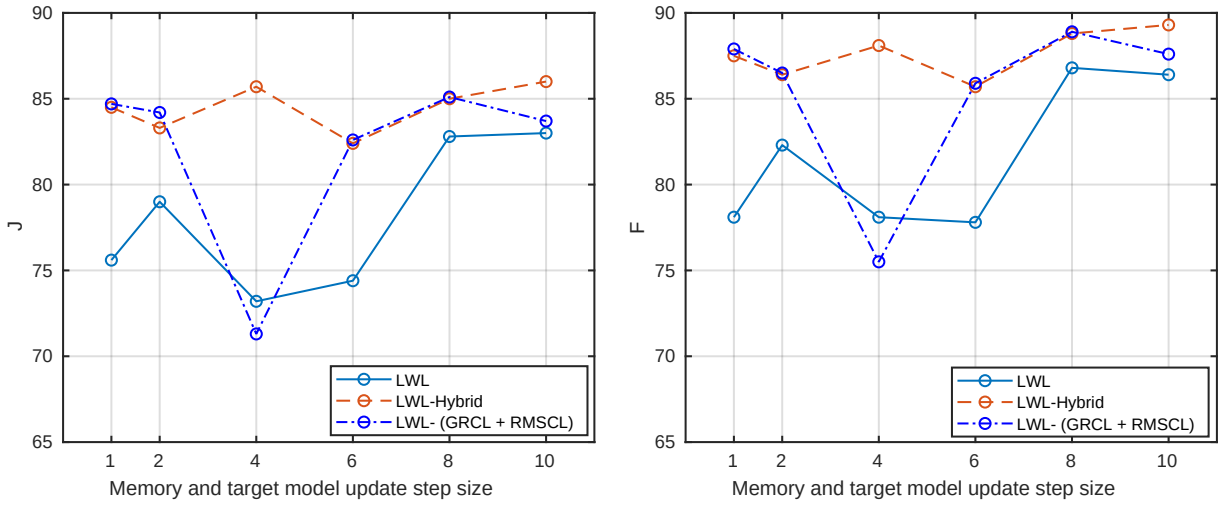

Figure 7.15: Quantitative evaluation ($\mathcal{J}$ left, $\mathcal{F}$ right) of the proposed Hybrid with a simple fusion of RMSCL and GRCL (GRCL + RMSCL) on the Long Videos dataset [1]. As seen, LWL-Hybrid with the message-passing mechanism between its RMSCL and GRCL parts has better performance than a simple fusion of RMSCL and GRCL LWL-(GRCL + RMSCL). Both proposed methods have better performance than LWL.

7.5.4 Hybrid

The most difficult aspect of developing the proposed Hybrid method is its hyper-parameter tuning of its two thresholds (τ and β). These two thresholds are the most important parameters of the Hybrid method's message-passing module. Following that, a comparison between the Hybrid method's performance with and without the message-passing module is conducted. Additionally, the influence of Hybrid's hyper-parameters is also discussed next.

Message-Passing Module between GRCL and RMSCL

The proposed Hybrid method with its message-passing module between GRCL and RMSCL was explained in Section 6.1. The message-passing module has two important hyper-parameter which are the selected threshold on the reconstruction error τ and the threshold β on the number of selected samples in the working memory. Figure 7.15 shows the comparison between LWL-Hybrid with and LWL-Hybrid without the message-passing module,

which is just the simple usage of RMSCL and GRCL (LWL-(GRCL+RMSCL)). As shown in Figure 7.15, there is a big performance degradation for LWL-(GRCL+RMSCL) on $\Delta_{\mathcal{M}} = \Delta_{\mathrm{C}} = 4$ and it is GRCL that amplifies and propagates the effect of segmentation errors which are placed in the memory $\mathcal{M}$ and are selected by RMSCL for updating the target model. This situation was discussed in Section 7.5.1 and Figure 7.10; however, this time the wrong segmented frame's information is selected by RMSCL and placed in the working memory. The proposed message-passing in LWL-Hybrid tackles this problem by not remembering all of the preceding continuously target model updates and giving the target model a chance to recover from remembering a wrong update. When a wrong target object is learned in an update and there are many visually similar adjacent frames in a video, RMSCL could select the same wrong set of samples from memory, which could amplify the effect of the wrong updated parameters. This problem can be handled by not updating the gated-regularizer memory when the selected samples for update are very similar when there are less than 3 selected samples for reconstructing the pooled version of the current sample X_{t+1}. The intuition behind the selected hyper-parameters of Hybrid is to avoid remembering the updates in which RMSCL is very confident. This situations happens when a set of selected samples with a small size (less than 3) is able to linearly reconstruct the current sample X_{t+1} using the estimated coefficients Ψ^t with a small reconstruction error (less than 2000).

7.5.5 Discussion

In [101], the authors explain theoretically why a likelihood-focused (replay-based) strategy is generally better than a prior-focused (regularization-based) solution. However, for an Online VOS method, based on the input video sequence, different situation could happen and the proposed Hybrid solution is the most general and efficient solution for Online VOS on long videos. There are also some concerns about the proposed Hybrid approach which are its computational complexity and the challenging hyper-parameters tuning. Additionally, Hybrid degrade the performance of Online VOS more than GRCL on short video datasets. It is worth noting that in Section 7.4, it was shown that the proposed methods do not have a significant negative or positive effect on the performance of baseline methods on short video sequences.

Another important benefits of the proposed approaches is that the offline trained parts of the adopted methods (LWL and JOINT) which are the encoder, decoder D, and the label encoder E, do not need to be re-trained for each proposed method. This makes it possible to apply the proposed methods on any Online VOS that follows the explained general Online VOS in Figure 3.1 and Algorithm 1 without any fine-tuning process.

Finally, it is important to mention again that the proposed methods are solving Equation 3.5; however, the focus of the optimizer algorithm is altered by focusing on each term of Bayesian equation by regularizing the model parameters (prior-focused) in GRCL or by changing the distribution of data in the memory (likelihood-focused) in RMSCL, and considering both in Hybrid.

Chapter 8

Conclusion

In this thesis, we proposed three novel methods, Gated-Regularizer Continual Learning (GRCL), Reconstruction-based Memory Selection Continual Learning (RMSCL), and Hybrid, that can be integrated with any Online VOS algorithm to improve its performance on long video sequences. The proposed approaches in this thesis can improve the capability of any Online VOS framework by making it more memory efficient and accurate in terms of performance. Furthermore, it is demonstrated that the proposed Hybrid method improves the robustness of the augmented baselines while increasing their computational complexity. The experimental results of this thesis show that the proposed approaches improved the accuracy of two baseline approaches (LWL [16] and JOINT [17]) on the Long Videos dataset, but did not significantly improve the baselines' performance on short video datasets (DAVIS16 [86], DAVIS17 [87], and YouTube-VOS18 [88]).

8.1 Summary of Contributions

In this thesis, continual learning was addressed in video object segmentation (VOS) leading to the following contributions:

- In Chapter 3, a general framework for Online VOS was presented, allowing the video object segmentation (VOS) challenge on long video sequences with concept drift to be formulated as a continual learning problem. The role of the target model C was highlighted in the problem formulation as a critical component for the adoption of the proposed continuous learning-inspired solutions (GRCL, RMSCL, and Hybrid).

- The proposed Gated Regularization-based Continual Learning (GRCL) approach (Chapter 4) is able to add a regularization term to any Online VOS loss function and regularizes important parameters associated with previous update steps. Another contribution of GRCL was its dynamic gated-regularizer memory $\mathcal{M}_W$, which is adjusted based on the number of the target model's regularized parameters.

- This thesis also contributed the Reconstruction-based Memory Selection Continual Learning (RMSCL) method explained in Chapter 5 for Online VOS models. The proposed likelihood-focused approach can be integrated into Online VOS to increase its speed and accuracy on long video sequences. The main challenge in the proposed RMSCL was to take the computational complexity of memory selection into account in order to have the most efficient memory selection for Online VOS. RMSCL decreases the computational complexity of Online VOS by providing smaller working memory $\mathcal{M}_W$ for each update step in comparison to Online VOS that updates the target model C on $\mathcal{M}$. Additionally, RMSCL re-weights the selected samples in $\mathcal{M}_W$ with its calculated coefficients Ψ which could bring a selected old sample from memory $\mathcal{M}$ bring back to the attention of the updating step.

- The Hybrid approach developed in Chapter 6, which combines RMSCL and GRCL, was the thesis's fourth contribution. The most important contribution of Hybrid was the message-passing mechanism between its RMSCL and GRCL parts, in which RMSCL tells GRCL when to remember the important parameter of the update step and when to ignore preserving the important parameters of an update step. The primary goal of the Hybrid method is to increase the robustness and accuracy of the fusion of RMSCL and GRCL; however, it increases the computational complexity of Online VOS.

8.2 Limitations and Future works

There are some limitations for each of the mentioned contributions to the thesis, which can be addressed in future work.

Continual Learning Formulation for Video Analysis

The first contribution of this thesis was addressing VOS from a continual learning perspective. The limitation for this problem formulation is that recently, matching-based VOS methods have become the state-of-the-art in terms of accuracy and speed on short and long videos; however, they usually use no online learning in their process. Thus, the first future work of this thesis is to combine matching-based methods with the online learning-based method and apply the proposed methods to provide a state-of-the-art semi-supervised VOS solution.

Additionally, the encode and decoder D part of the evaluated baseline methods (LWL and JOINT) were trained offline on short video VOS datasets; however, one future work of this study is to fine-tune the encoder and decoder parts of general Online VOS explained in Section 3.1 to be able to get better results on long video sequences. The concept behind this future work is that there could be some better features that encoders could provide specifically for long video sequences, which is partly addressed in ISVOS [85].

As is shown in Section 7.4, the proposed methods of this thesis do not have many impacts on short video datasets and there are not many publicly available long video VOS datasets. Recently, CLVOS23 [102] released to extend the Long Videos dataset [1] to 9 videos and 281 labelled frames for evaluation. Applying and refining the proposed methods suggested in this thesis to CLVOS23 is planned for the future.

Another potential future work of this thesis is to expand the proposed method to other areas of video processing, specifically object tracking, which could suffer from forgetting during the online learning process of the online tracking.

Gated-Regularizer Continual Learning (GRCL)

In GRCL, all the hyper-parameters are tuned for the evaluated Online VOS baselines on short and long VOS datasets; however, training the target model for more epochs alongside re-traninig the encoder and decoder can show a bigger impact of GRCL since the regularization is more effective with better training. It is worth noting that for all the

proposed methods, the goal was to add the proposed approach to the baselines without any need for change to the structure or the hyper-parameters of the baseline.

The main limitation of GRCL is that when the prior knowledge learned during the preceding update time is wrong, as discussed in Section 7.5.1 and Figure 7.10, GRCL insists on remembering the wrong prior (learned information) for the next update steps, which causes the model to lose the correct target object and cannot be recovered from the wrong information in the memory. Considering this issue, another future work of this thesis is to propose a solution (using heuristic data) to define a quality metric for the available data in the memory as an indicator for remembering a target model update step.

Reconstruction-based Memory Selection Continual Learning (RMSCL)

The memory selection method in RMSCL uses the input frame features in Equation (5.3); however, the features are used for segmenting the object. In other words, the feature selection could be more accurate if the features were combined with the object information for doing the reconstruction in Equation (5.3). Thus, one of the future tasks for RMSCL would be finding more related sets of features that is provided by RMSCL and related weights (Ψ^t) for updating the target model before segmenting the current feature X_{t+1} .

Hybrid Approach

The main limitation of the Hybrid approach is its hyper-parameter tuning, which makes it difficult to generalize to all situations. The hyper-parameters should be the same for long and short video sequences, indicating the method does not have prior knowledge about the length of the video. In the proposed Hybrid method, the message-passing between RMSCL and GRCL is done in one way, which means RMSCL provides some information for GRCL to update its memory; however, the impact of GRCL on RMSCL is not considered. In future work, we can consider having a stronger two-way connection between RMSCL and GRCL for a better Hybrid method.

Additionally, the selected thresholds of Hybrid (β and τ) are static; however, online learning is dynamic. The last future work of this thesis is to set dynamic thresholds for Hybrid method, which could make the proposed Hybrid method more generalized the the proposed Hybrid method in this thesis.

References

[1] Yongqing Liang, Xin Li, Navid Jafari, and Jim Chen. Video object segmentation with adaptive feature bank and uncertain-region refinement. *Advances in Neural Information Processing Systems*, 33:3430–3441, 2020.

[2] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[3] Roger Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97(2):285, 1990.

[4] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[5] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[6] Rahaf Aljundi. *Continual Learning in Neural Networks*. PhD thesis, KU Leuven, 2019.

[7] Alexander Gepperth and Barbara Hammer. Incremental learning algorithms and applications. 2016.

[8] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. *Computing Research Repository (CoRR)*, abs/1905.13260, 2019.

[9] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):1–37, 2014.

[10] Zeki Erdem, Robi Polikar, Fikret Gurgen, and Nejat Yumusak. Ensemble of svms for incremental learning. In *International Workshop on Multiple Classifier Systems*, pages 246–256. Springer, 2005.

[11] Hieu Pham, Melody Y Guan, Barret Zoph, Quoc V Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. *arXiv preprint arXiv:1802.03268*, 2018.

[12] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[13] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] 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, 2009.

[16] Goutam Bhat, Felix Järemo Lawin, Martin Danelljan, Andreas Robinson, Michael Felsberg, Luc Van Gool, and Radu Timofte. Learning what to learn for video object segmentation. In *European Conference on Computer Vision*, pages 777–794. Springer, 2020.

[17] Yunyao Mao, Ning Wang, Wengang Zhou, and Houqiang Li. Joint inductive and transductive learning for video object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9670–9679, 2021.

[18] Preksha Pareek and Ankit Thakkar. A survey on video-based human action recognition: recent updates, datasets, challenges, and applications. *Artificial Intelligence Review*, 54:2259–2322, 2021.

[19] Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip HS Torr. Fast online object tracking and segmentation: A unifying approach. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 1328–1338, 2019.

[20] 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 International Conference on Computer Vision*, pages 9226–9235, 2019.

[21] Ho Kei Cheng and Alexander G Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *European Conference on Computer Vision*, pages 640–658. Springer, 2022.

[22] Andreas Robinson, Felix Jaremo Lawin, Martin Danelljan, Fahad Shahbaz Khan, and Michael Felsberg. Learning fast and robust target models for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7406–7415, 2020.

[23] Yu Liu, Lingqiao Liu, Haokui Zhang, Hamid Rezatofighi, Qingsen Yan, and Ian Reid. Meta learning with differentiable closed-form solver for fast video object segmentation. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8439–8446. IEEE, 2020.

[24] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler L Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[25] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*, pages 11816–11825, 2019.

[26] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278–2324, 1998.

[27] German I. Parisi and Vincenzo Lomonaco. Online continual learning on sequences. *Studies in Computational Intelligence*, page 197–221, 2020.

[28] Jyrki Kivinen, Alexander J Smola, and Robert C Williamson. Online learning with kernels. *IEEE transactions on signal processing*, 52(8):2165–2176, 2004.

[29] James A Anderson. *An introduction to neural networks*. MIT press, 1995.

[30] Berndt Müller, Joachim Reinhardt, and Michael T Strickland. *Neural networks: an introduction*. Springer Science & Business Media, 1995.

[31] Anil K Jain, Jianchang Mao, and K Moidin Mohiuddin. Artificial neural networks: A tutorial. *Computer*, 29(3):31–44, 1996.

[32] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[33] Waseem Rawat and Zenghui Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation*, 29(9):2352–2449, 2017.

[34] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[35] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[36] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[37] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[38] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[39] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[40] Shai Shalev-Shwartz et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

[41] Steven CH Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. Online learning: A comprehensive survey. *arXiv preprint arXiv:1802.02871*, 2018.

[42] Oren Anava, Elad Hazan, and Shie Mannor. Online learning for adversaries with memory: price of past mistakes. *Advances in Neural Information Processing Systems*, 28, 2015.

[43] Yen-Chang Hsu, Yen-Cheng Liu, and Zsolt Kira. Re-evaluating continual learning scenarios: A categorization and case for strong baselines. *arXiv preprint arXiv:1810.12488*, 2018.

[44] Sebastian Farquhar and Yarin Gal. A unifying bayesian view of continual learning. *arXiv preprint arXiv:1902.06494*, 2019.

[45] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017.

[46] Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733*, 2018.

[47] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 2019.

[48] Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. *arXiv preprint arXiv:1805.06370*, 2018.

[49] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154, 2018.

[50] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.

[51] Jose M Alvarez and Mathieu Salzmann. Learning the number of neurons in deep networks. In *Advances in Neural Information Processing Systems*, pages 2270–2278, 2016.

[52] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in neural information processing systems*, pages 2074–2082, 2016.

[53] Robert M French. Using semi-distributed representations to overcome catastrophic forgetting in connectionist networks. In *Proceedings of the 13th annual cognitive science society conference*, pages 173–178, 1991.

[54] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, pages 2990–2999, 2017.

[55] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[56] JaeHong Yoon, Jeongtae Lee, Eunho Yang, and Sung Ju Hwang. Lifelong learning with dynamically expandable network. In *International Conference on Learning Representations*. International Conference on Learning Representations, 2018.

[57] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[58] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[59] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, pages 6467–6476, 2017.

[60] David Isele and Akansel Cosgun. Selective experience replay for lifelong learning. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[61] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and M Ranzato. Continual learning with tiny episodic memories. 2019.

[62] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[63] Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In *Conference on Robot Learning*, pages 17–26, 2017.

[64] Rui Yao, Guosheng Lin, Shixiong Xia, Jiaqi Zhao, and Yong Zhou. Video object segmentation and tracking: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(4):1–47, 2020.

[65] Sergi Caelles, Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. One-shot video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 221–230, 2017.

[66] Federico Perazzi, Anna Khoreva, Rodrigo Benenson, Bernt Schiele, and Alexander Sorkine-Hornung. Learning video object segmentation from static images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2663–2672, 2017.

[67] Zhishan Zhou, Lejian Ren, Pengfei Xiong, Yifei Ji, Peisen Wang, Haoqiang Fan, and Si Liu. Enhanced memory network for video segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[68] Yuan-Ting Hu, Jia-Bin Huang, and Alexander Schwing. Maskrnn: Instance level video object segmentation. *Advances in neural information processing systems*, 30, 2017.

[69] Carles Ventura, Miriam Bellver, Andreu Girbau, Amaia Salvador, Ferran Marques, and Xavier Giro-i Nieto. Rvos: End-to-end recurrent network for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5277–5286, 2019.

[70] Paul Voigtlaender and Bastian Leibe. Online adaptation of convolutional neural networks for video object segmentation. In *British Machine Vision Conference 2017, BMVC 2017, London, UK, September 4-7, 2017*. BMVA Press, 2017.

[71] Li Hu, Peng Zhang, Bang Zhang, Pan Pan, Yinghui Xu, and Rong Jin. Learning position and target consistency for memory-based video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4144–4154, 2021.

[72] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Shengping Zhang, and Wenxiu Sun. Efficient regional memory network for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1286–1295, 2021.

[73] Ho Kei Cheng, Yu-Wing Tai, and Chi-Keung Tang. Rethinking space-time networks with improved memory coverage for efficient video object segmentation. *Advances in Neural Information Processing Systems*, 34:11781–11794, 2021.

[74] Zongxin Yang, Yunchao Wei, and Yi Yang. Associating objects with transformers for video object segmentation. *Advances in Neural Information Processing Systems*, 34:2491–2502, 2021.

[75] Fanchao Lin, Hongtao Xie, Yan Li, and Yongdong Zhang. Query-memory re-aggregation for weakly-supervised video object segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2038–2046, 2021.

[76] Yong Liu, Ran Yu, Jiahao Wang, Xinyuan Zhao, Yitong Wang, Yansong Tang, and Yujiu Yang. Global spectral filter memory network for video object segmentation. In *European Conference on Computer Vision*, pages 648–665. Springer, 2022.

[77] Hongje Seong, Junhyuk Hyun, and Euntai Kim. Kernelized memory network for video object segmentation. In *European Conference on Computer Vision*, pages 629–645. Springer, 2020.

[78] Hongje Seong, Seoung Wug Oh, Joon-Young Lee, Seongwon Lee, Suhyeon Lee, and Euntai Kim. Hierarchical memory matching network for video object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12889–12898, 2021.

[79] Xiaoxiao Li and Chen Change Loy. Video object segmentation with joint re-identification and attention-aware mask propagation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 90–105, 2018.

[80] Yi-Hsuan Tsai, Ming-Hsuan Yang, and Michael J Black. Video segmentation via object flow. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3899–3908, 2016.

[81] Linjie Yang, Yanran Wang, Xuehan Xiong, Jianchao Yang, and Aggelos K Katsaggelos. Efficient video object segmentation via network modulation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6499–6507, 2018.

[82] Steven S. Beauchemin and John L. Barron. The computation of optical flow. *ACM computing surveys (CSUR)*, 27(3):433–466, 1995.

[83] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9481–9490, 2019.

[84] Jingchun Cheng, Yi-Hsuan Tsai, Shengjin Wang, and Ming-Hsuan Yang. Segflow: Joint learning for video object segmentation and optical flow. In *Proceedings of the IEEE international conference on computer vision*, pages 686–695, 2017.

[85] Junke Wang, Dongdong Chen, Zuxuan Wu, Chong Luo, Chuanxin Tang, Xiyang Dai, Yucheng Zhao, Yujia Xie, Lu Yuan, and Yu-Gang Jiang. Look before you match: Instance understanding matters in video object segmentation. *arXiv preprint arXiv:2212.06826*, 2022.

[86] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 724–732, 2016.

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

[88] Ning Xu, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. Youtube-vos: A large-scale video object segmentation benchmark. *arXiv preprint arXiv:1809.03327*, 2018.

[89] Yu Li, Zhuoran Shen, and Ying Shan. Fast video object segmentation using the global context module. In *European Conference on Computer Vision*, pages 735–750. Springer, 2020.

[90] 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.

[91] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[92] Robert Bassett and Julio Deride. Maximum a posteriori estimators as a limit of bayes estimators. *Mathematical Programming*, 174:129–144, 2019.

[93] Robert Hecht-Nielsen. Theory of the backpropagation neural network. In *Neural networks for perception*, pages 65–93. Elsevier, 1992.

[94] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[95] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Ng. Efficient sparse coding algorithms. *Advances in neural information processing systems*, 19, 2006.

[96] George EP Box and George C Tiao. *Bayesian inference in statistical analysis*. John Wiley & Sons, 2011.

[97] Vincenzo Lomonaco, Lorenzo Pellegrini, Gabriele Graffieti, and Davide Maltoni. Architect, regularize and replay (arr): a flexible hybrid approach for continual learning. *arXiv preprint arXiv:2301.02464*, 2023.

[98] Seoung Wug Oh, Joon-Young Lee, Kalyan Sunkavalli, and Seon Joo Kim. Fast video object segmentation by reference-guided mask propagation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7376–7385, 2018.

[99] Hirotugu Akaike. A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723, 1974.

[100] Ken Aho, DeWayne Derryberry, and Teri Peterson. Model selection for ecologists: the worldviews of aic and bic. *Ecology*, 95(3):631–636, 2014.

[101] Jeremias Knoblauch, Hisham Husain, and Tom Diethe. Optimal continual learning has perfect memory and is np-hard. *arXiv preprint arXiv:2006.05188*, 2020.

[102] Amir Nazemi, Zeyad Moustafa, and Paul Fieguth. Clvos23: A long video object segmentation dataset for continual learning. *arXiv preprint arXiv:2304.04259*, 2023.

[103] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[104] James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

[105] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 532–547, 2018.

[106] Jacob MJ Murre. *Learning and categorization in modular neural networks*. Psychology Press, 2014.

[107] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[108] Ryne Roady, Tyler L Hayes, Hitesh Vaidya, and Christopher Kanan. Stream-51: Streaming classification and novelty detection from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 228–229, 2020.

[109] Jonathon Luiten, Paul Voigtlaender, and Bastian Leibe. Premvos: Proposal-generation, refinement and merging for video object segmentation. In *Asian Conference on Computer Vision*, pages 565–580. Springer, 2018.

[110] Shuangjie Xu, Daizong Liu, Linchao Bao, Wei Liu, and Pan Zhou. Mhp-vos: Multiple hypotheses propagation for video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 314–323, 2019.

[111] Yanchao Yang, Antonio Loquercio, Davide Scaramuzza, and Stefano Soatto. Unsupervised moving object detection via contextual information separation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 879–888, 2019.

[112] Wenguan Wang, Hongmei Song, Shuyang Zhao, Jianbing Shen, Sanyuan Zhao, Steven CH Hoi, and Haibin Ling. Learning unsupervised video object segmentation through visual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3064–3074, 2019.

[113] 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.

[114] Xiaohui Zeng, Renjie Liao, Li Gu, Yuwen Xiong, Sanja Fidler, and Raquel Urtasun. Dmm-net: Differentiable mask-matching network for video object segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3929–3938, 2019.

[115] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. Continual learning: A comparative study on how to defy forgetting in classification tasks. *arXiv preprint arXiv:1909.08383*, 2019.

[116] Wen Wei and Jerry M Mendel. Maximum-likelihood classification for digital amplitude-phase modulations. *IEEE transactions on Communications*, 48(2):189–193, 2000.

[117] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. 2014.

[118] John Duchi. Derivations for linear algebra and optimization. *Berkeley, California*, 3:2325–5870, 2007.

[119] P. Siva and A. Wong. Author guidelines for journal of computational vision and imaging systems. *Journal of Computational Vision and Imaging Systems*, 2015.

[120] P. Siva and A. Wong. More author guidelines. In *Proc. CVIS*, pages 1–3, 2015.

[121] 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.

[122] German I Parisi and Vincenzo Lomonaco. Online continual learning on sequences. *Recent Trends in Learning From Data*, pages 197–221, 2020.

[123] Fanqing Lin, Yao Chou, and Tony Martinez. Flow adaptive video object segmentation. *Image and Vision Computing*, 94:103864, 2020.

[124] Ziqin Wang, Jun Xu, Li Liu, Fan Zhu, and Ling Shao. Ranet: Ranking attention network for fast video object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3978–3987, 2019.

[125] Lu Zhang, Zhe Lin, Jianming Zhang, Huchuan Lu, and You He. Fast video object segmentation via dynamic targeting network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5582–5591, 2019.

[126] Haochen Wang, Xiaolong Jiang, Haibing Ren, Yao Hu, and Song Bai. Swiftnet: Real-time video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1296–1305, 2021.

[127] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. Incremental learning of object detectors without catastrophic forgetting. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3400–3409, 2017.

[128] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

[129] Yunyao Mao, Ning Wang, Wengang Zhou, and Houqiang Li. Joint inductive and transductive learning for video object segmentation. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 9650–9659. IEEE, 2021.

[130] Yanchao Yang, Brian Lai, and Stefano Soatto. Dystab: Unsupervised object segmentation via dynamic-static bootstrapping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2826–2836, 2021.

[131] Guangyuan Shi, Jiaxin Chen, Wenlong Zhang, Li-Ming Zhan, and Xiao-Ming Wu. Overcoming catastrophic forgetting in incremental few-shot learning by finding flat minima. *Advances in Neural Information Processing Systems*, 34:6747–6761, 2021.

[132] Huihui Liu, Yiding Yang, and Xinchao Wang. Overcoming catastrophic forgetting in graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8653–8661, 2021.

[133] Kuluhan Binici, Nam Trung Pham, Tulika Mitra, and Karianto Leman. Preventing catastrophic forgetting and distribution mismatch in knowledge distillation via synthetic data. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 663–671, 2022.

[134] Pei-Hung Chen, Wei Wei, Cho-Jui Hsieh, and Bo Dai. Overcoming catastrophic forgetting by bayesian generative regularization. In *International Conference on Machine Learning*, pages 1760–1770. PMLR, 2021.

[135] Pauching Yap, Hippolyt Ritter, and David Barber. Addressing catastrophic forgetting in few-shot problems. In *International Conference on Machine Learning*, pages 11909–11919. PMLR, 2021.

[136] Fan Zhou and Chengtai Cao. Overcoming catastrophic forgetting in graph neural networks with experience replay. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4714–4722, 2021.

[137] Craig Atkinson, Brendan McCane, Lech Szymanski, and Anthony Robins. Pseudo-rehearsal: Achieving deep reinforcement learning without catastrophic forgetting. *Neurocomputing*, 428:291–307, 2021.

[138] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248, 2018.

[139] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 374–382, 2019.

[140] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13208–13217, 2020.

[141] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

[142] Sang-Woo Lee, Chung-Yeon Lee, Dong-Hyun Kwak, Jiwon Kim, Jeonghee Kim, and Byoung-Tak Zhang. Dual-memory deep learning architectures for lifelong learning of everyday human behaviors. In *IJCAI*, pages 1669–1675, 2016.

[143] MTCAJ Thomas and A Thomas Joy. *Elements of information theory*. Wiley-Interscience, 2006.

[144] Ling Jian, Jundong Li, Kai Shu, and Huan Liu. Multi-label informed feature selection. In *IJCAI*, volume 16, pages 1627–33, 2016.

[145] Jundong Li and Huan Liu. Challenges of feature selection for big data analytics. *IEEE Intelligent Systems*, 32(2):9–15, 2017.

[146] Jundong Li, Xia Hu, Jiliang Tang, and Huan Liu. Unsupervised streaming feature selection in social media. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1041–1050, 2015.

[147] Jundong Li, Xia Hu, Ling Jian, and Huan Liu. Toward time-evolving feature selection on dynamic networks. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 1003–1008. IEEE, 2016.

[148] Xiaokai Wei and S Yu Philip. Unsupervised feature selection by preserving stochastic neighbors. In *Artificial intelligence and statistics*, pages 995–1003. PMLR, 2016.

[149] Jundong Li, Jiliang Tang, and Huan Liu. Reconstruction-based unsupervised feature selection: An embedded approach. In *IJCAI*, pages 2159–2165, 2017.

[150] Rui Zhang and Xuelong Li. Unsupervised feature selection via data reconstruction and side information. *IEEE Transactions on Image Processing*, 29:8097–8106, 2020.

[151] Yirong Wu, Qi Ren, Shuifa Sun, and Tinglong Tang. Memory reconstruction based dual encoders for anomaly detection. In *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2244–2250. IEEE, 2022.

[152] Yanfang Liu, Dongyi Ye, Wenbin Li, Huihui Wang, and Yang Gao. Robust neighborhood embedding for unsupervised feature selection. *Knowledge-based systems*, 193:105462, 2020.

[153] Dorothy M Greig, Bruce T Porteous, and Allan H Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society: Series B (Methodological)*, 51(2):271–279, 1989.

[154] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.