Towards Image and Video Understanding in Challenging Scenarios

Thesis approved by
the Department of Computer Science
University of Kaiserslautern-Landau
for the award of the Doctoral Degree
Doctor of Engineering (Dr.-Ing.)

to

Fatemeh Azimi

Date of Defense:	09.12.2024
Dean:	Prof. Dr. Christoph Garth
Reviewer:	Prof. Dr. Prof. h.c. Andreas Dengel
Reviewer:	Prof. Dr. Didier Stricker

DE-386

RP TU Rheinland-Pfälzische Technische Universität Kaiserslautern Landau

Abstract

Learning-based solutions have revolutionized the field of Artificial Intelligence (AI), pushing it to new frontiers in a variety of domains. AI advancements owe much to highly curated datasets that enable the training and testing of complex deep learning models, leading to excellent accuracy in controlled academic environments. However, it is essential to acknowledge that the efficacy of these models in these lab settings does not fully encapsulate the complexities and challenges encountered in real-life applications. To ensure the practical applicability of learning-based solutions, it is crucial to understand their limitations and performance under diverse and challenging measurements. This requires bridging the gap between academic benchmarks and the complexities of the real world.

In this thesis, we take a step toward better understanding the limitations of current deep learning models in handling corner cases and challenging scenarios, with a focus on computer vision tasks across image and video domains. Through this exploration, we identify areas and ways to improve the robustness of computer vision solutions.

In the image domain, we study image classification and analyze the model behavior when processing images containing background noise and clutter. We extend this study to investigate salient image classification, a scenario in which multiple objects are present in a scene, and the model is expected to classify the most prominent or salient one.

In the video domain, we explore the fundamental task of spatiotemporal feature correspondence learning. This task has diverse applications, such as video object segmentation and tracking. Our investigation delves into challenging scenarios, including tracking smaller objects, managing occlusion, coping with crowded scenes, and efficiently processing longer videos. Furthermore, we investigate self-supervised learning methods for spatiotemporal correspondence learning, motivated by the high cost of annotating video data for this task and the challenges that arise when training on small datasets. Finally, we study the problem of out-of-domain generalization on video data, a critical issue that affects the applicability of learning-based solutions. To this end, we evaluate several ways for using self-supervised learning to mitigate the adverse effects of domain shift, enabling the model to perform well in new, unseen domains.

We hope this work fosters advancements in the field of AI by providing insights and directions for designing more robust models that deliver enhanced performance in diverse and complex scenarios.

Acknowledgement

I would like to express my sincere gratitude and appreciation to my supervisor Professor Dengel, for granting me this invaluable opportunity and for his consistent support and guidance throughout the entire process of my Ph.D. His expertise, mentorship, and encouragement have been instrumental in shaping my academic and research journey.

I sincerely thank my dear colleagues and team leads, Federico Raue, Joern Hees, and Sebastian Palacio, for their tireless support, valuable mentorship, and creating a truly supportive and inspiring environment. However, their impact extends beyond the realm of work. Beyond being exceptional professionals, they have also become true friends. Their genuine care and compassion have created an uplifting and inspiring environment that I am deeply grateful for.

To my loving partner, Tewodros, I am deeply grateful for your kindness, patience, and support. Your presence and encouragement have been a constant source of strength and motivation, and I am truly blessed to have you by my side throughout this journey.

I would like to thank my dear cousins, Malihe and Reza, who have been a continuous source of inspiration, and I have always relied on their support. I dearly thank Nasrin, Malihe, Neda, and Maryam, my dear cousins who, despite the physical distance between us, have always enveloped me with the warmth and love of a family. Your constant presence in my life has been a priceless source of comfort and I am truly appreciative.

To my dear friends, Nazli, Annette, Spideh, Peyman, Dena, Sahar, Soheil, Shahrzad, Faeze, and Shideh, I am deeply grateful for your friendship. You have always lifted me up when I was down and generously stood by my side, providing me with encouragement, understanding, and a sense of belonging. Your presence in my life has been an immense source of joy, and I hold each of you dearly in my heart.

I am tremendously grateful to my mom, who has consistently encouraged me to pursue excellence in education and has wholeheartedly supported every ambition of mine, even in the face of societal norms prevalent in the city of my birth. Likewise, I extend my heartfelt gratitude to my dad, whose love radiates as a guiding light in my life, filling my heart with warmth and affection. His constant presence, encouragement, and support have made me who I am as a person. It is with deep gratitude that I dedicate this thesis to my mom and dad, as they are the pillars of my life and to whom I owe everything.

2 Publications

This thesis builds upon the following peer-reviewed conference publications.

- **Chapter 3:** Azimi, Fatemeh, Federico Raue, Jörn Hees, and Andreas Dengel. "A reinforcement learning approach for sequential spatial transformer networks." In *International Conference on Artificial Neural Networks*, pp. 585–597. Springer, 2019.

- **Chapter 3:** Azimi, Fatemeh, Jean-Francois Jacques Nicolas Nies, Sebastian Palacio, Federico Raue, Jörn Hees, and Andreas Dengel. "Spatial Transformer Networks for Curriculum Learning." In *The International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, 2022.

- **Chapter 3:** Azimi, Fatemeh, David Dembinsky, Federico Raue, Jörn Hees, Sebastian Palacio, and Andreas Dengel. "Sequential Spatial Transformer Networks for Salient Object Classification." In *12th International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, 2023.

- **Chapter 4:** Azimi, Fatemeh, Benjamin Bischke, Sebastian Palacio, Federico Raue, Jörn Hees, and Andreas Dengel. "Revisiting sequence-to-sequence video object segmentation with multi-task loss and skip-memory." In *25th International Conference on Pattern Recognition (ICPR)*, pp. 5376–5383, 2021.

- **Chapter 4:** Azimi, Fatemeh, Stanislav Frolov, Federico Raue, Jörn Hees, and Andreas Dengel. "Hybrid-S2S: Video Object Segmentation with Recurrent Networks and Correspondence Matching." In *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2021)*, pp. 182–192. SCITEPRESS, 2021.

- **Chapter 4:** Azimi, Fatemeh, Federico Raue, Jörn Hees, and Andreas Dengel. "Rethinking RNN-based Video Object Segmentation." In *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2021)*, 2021.

- **Chapter 5:** Azimi, Fatemeh, Fahim Mannan, and Felix Heide. "S^3Track: Self-supervised Multi-Object Tracking with Soft Assignment Flow." In *Proceedings of the ECCV Workshop*, 2024.

- **Chapter 5:** Azimi, Fatemeh, Sebastian Palacio, Federico Raue, Jörn Hees, Luca Bertinetto, and Andreas Dengel. “Self-supervised test-time adaptation on video data.” In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3439–3448, 2022.

Contents

3 Introduction

Artificial Intelligence (AI), the art of creating machines that can reason similar to the human cognitive system, is a long-standing ambition that goes back centuries to centuries ago when philosophers like Aristotle were hypothesizing about the workings of the human mind and the systematic way of thinking. However, the official establishment of the AI field known to us today did not happen until the summer of 1956 during the Dartmouth Workshop, the first academic gathering where researchers from various areas came together to explore the possibilities of creating machines that could exhibit intelligent behavior, marking a turning point in the history of AI [Moo06; MC04]. Led by John McCarthy, Marvin Minsky, Nathaniel Rochester, and Claude Shannon, participants in this workshop formulated early AI concepts and proposed ambitious goals, setting the stage for decades of exploration, innovation, and advancements in AI technologies [McC+06].

Following this event, the field of AI witnessed the emergence of different approaches and viewpoints, including symbolic AI. This paradigm, also known as classical or traditional AI, focused on representing knowledge and performing reasoning using symbolic logic and formal rules [Lev86; Nil82]. Aiming to create intelligent systems by explicitly encoding human knowledge and rules in the form of logical relationships, symbolic AI researchers developed expert systems, knowledge-based systems, and rule-based systems that employed symbolic representations and inference mechanisms to solve problems in specific domains [Lug05]. These methods enjoyed initial success, with expert systems being widely deployed in various industries. However, in the 1980s and early 1990s, the field experienced a period known as the AI winter. During this time, the limitations of symbolic AI became evident as the reliance on manually crafted rules and the inability to learn from data and deal with complex real-world problems hindered further progress, leading to a decline in interest in the field [Nil10].

The resurgence of AI in the early 2010s was fueled by remarkable advancements in deep learning methods, which were made possible by significant progress in hardware, particularly the utilization of Graphics Processing Units (GPUs) and the availability of large-scale datasets such as ImageNet [Den+09; KSH12]. Over the past decade, deep learning methods such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Transformers have revolutionized the field of AI, demonstrating exceptional performance

across various tasks, surpassing traditional AI methods by a wide margin [Wan+23; SVL14; Vas+17].

This pivotal shift in the AI landscape has prompted researchers to increasingly adopt data-driven approaches. As a result, within a relatively short span of about a decade, deep learning models have become the fundamental building blocks of numerous AI applications, driving breakthroughs in areas such as computer vision, natural language understanding, and autonomous systems. As examples of these unprecedented advancements, one can think of Reinforcement Learning (RL) techniques being successfully applied to develop AI systems capable of playing complex strategy games like Go, surpassing human performance. Another notable breakthrough is the development of large language models such as chatGPT [Bro+20] that can engage in interactive conversations and provide human-like responses across various topics. Moreover, in the realm of AI generation, generative algorithms like Diffusion models [HJA20] and CLIP [Rad+21] have exhibited extraordinary advancements, empowering the generation of highly realistic outputs in domains like image synthesis and text generation.

However, despite the remarkable advancements in deep learning methods and the significant role AI plays in our daily lives, there is still much progress to be made. One key limitation hindering the widespread applicability of these methods stems from their development and evaluation within controlled laboratory settings with a focus on optimizing performance scores on specialized datasets. Yet, there is a growing recognition that real-world scenarios can be significantly different, often presenting unique challenges that are not well-captured by existing benchmarks. For instance, factors like data bias, robustness, and out-of-domain generalization pose ongoing challenges that need to be addressed for deep learning solutions to be more reliable and trustworthy. These aspects motivate the progress of this thesis, as elaborated in the following.

3.1 Motivation

To truly rely on learning-based solutions, it is imperative to thoroughly analyze the behavior of these models beyond the performance scores on academic datasets and evaluations under controlled lab settings. In this thesis, we aim to uncover the weaknesses of existing deep learning methods in challenging real-world measurements, paving the way for more robust and reliable systems with a focus on computer vision applications. Undoubtedly, this is a formidable challenge that can be addressed from multiple, potentially orthogonal perspectives. As such, we embark on a journey toward better understanding the limitations of current approaches, hoping to take a step toward solving these issues.

This work investigates the limitations of deep learning solutions across multiple domains within the field of computer vision. In the image domain, our primary focus lies in image classification, a well-established task that serves as a standard benchmark for evaluating deep learning architectures. In the video domain, we investigate spatiotemporal correspondence learning—the crucial task of identifying the matching features across both spatial and temporal dimensions within video data. Establishing these correspondences unlocks various practical applications, such as Video Object Segmentation (VOS) and Multi-Object Tracking (MOT), illustrated in Figure 3.1. The learning of these correspondences is accomplished by training neural networks that can compute distinctive and representative image features, ensuring that features belonging to the same object instance are grouped closely together in the embedding space while those from different instances are well-separated.

Image Domain: Classification

In image classification, we investigate a scenario where images contain a high level of background noise and clutter. We note that the state-of-the-art classifiers have been developed using datasets such as CIFAR [KH+09] and ImageNet [Den+09] that exhibit high levels of curation in terms of containing unoccluded and prominent objects relatively located at the image center with minimal background noise. These datasets were meant to answer the question: can learning-based solutions distinguish between all object classes in the dataset? While these datasets have been crucial for advancing machine learning solutions, they do not fully represent the conditions encountered in natural image classification. This discrepancy raises a fundamental question: even if we successfully solve ImageNet, have we truly solved the challenges of image classification as encountered in real-world scenarios? This is an important question as the answer to that determines if we can reliably use current models in real-world scenarios, including various levels of noise, lighting conditions, and object sizes. These factors motivate conducting further research on methods that can effectively tackle these aspects.

Video Domain: Spatiotemporal Correspondence Learning

Video Object Segmentation (VOS). In the video domain, our exploration begins with the task of VOS. This task aims to densely track a set of target objects starting from the segmentation mask provided in the first frame. One of the prominent paradigms for addressing VOS is through propagation-based algorithms. These methods involve the continuous propagation of the object mask from one frame to the next, leveraging feature similarity and motion information. We start our investigation with a close examination of a state-of-the-art propagation-based method. We are particularly interested in this method as

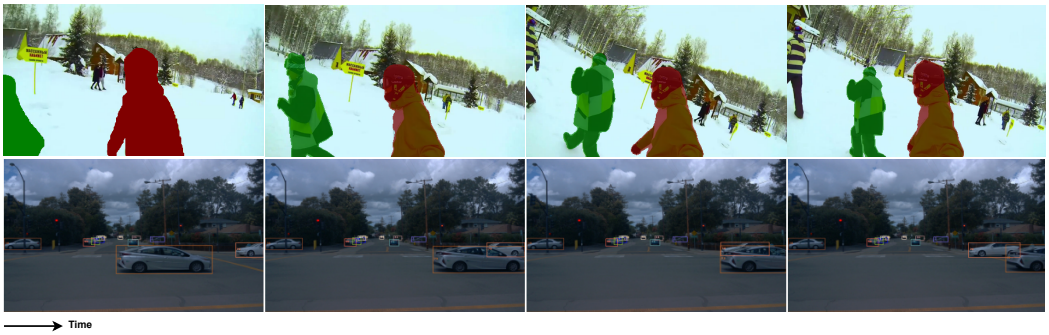


Fig. 3.1: Visual examples for spatiotemporal correspondence learning applications. The first and second rows showcase video object segmentation and multi-object tracking applications.

it leverages the power of RNNs, enabling the integration of both visual and motion cues within a single architecture and in an end-to-end manner. However, our analysis reveals that the model's performance deteriorates when tracking smaller objects and capturing fine object details around the edges. Furthermore, we observed a decline in accuracy for occluded objects and longer videos. It is important to note that these particular cases are relatively under-represented in current datasets [Pon+17; Xu+18]; as a result, the models may obtain a high accuracy without resolving these aspects. Nevertheless, it is crucial to address these issues to enhance the method's robustness and make it applicable in these challenging real-world settings.

Self-supervised Learning for Correspondence Learning. One of the known limitations of the learning-based solutions is dependence on labeled data. This concern becomes particularly pronounced in the context of spatiotemporal learning, where annotators must label every frame of a video with multiple object instances. Consequently, the scarcity and labor-intensive nature of obtaining such extensive annotations limits harnessing the full potential of these models. While under lab conditions and with small datasets, we have per-frame labels for a set of desired object categories, it becomes infeasible to label a sufficient volume of data covering diverse scenarios required for training large deep learning models. In this context, self-supervised solutions emerge as a crucial practical approach, especially for tasks demanding an abundance of spatiotemporal video labels, such as MOT.

Recently, there has been a surge in efforts to devise self-supervised methods tailored to spatiotemporal correspondence learning. These proposed solutions mainly center around proxy objectives based on color constancy assumptions across video frames. However, using color information as the supervision signal leads to shortcomings in crowded scenes featuring multiple similar objects with the same color. This is particularly problematic in the context of MOT, where traffic scenarios frequently involve vehicles sharing identical colors, rendering color information ineffective as a discriminative signal in such situations.

Therefore, further research in this domain is much needed to develop effective solutions and design more robust models.

Test Time Adaptation on Video Data. In the next part of this thesis, we delve into a longstanding challenge of out-of-domain generalization in deep learning models, focused on video data. It is well known that machine learning solutions only work well when training and testing data follow the same distribution. However, this requirement is frequently unattainable in real-world applications. For instance, training a VOS model on data captured in sunny weather may prove inadequate when facing a testing scenario comprising data samples from snowy weather. Clearly, attempting to finetune the model on the test data, where labels are likely unavailable, is not a feasible approach.

Several methods have been recently proposed in the image domain under the name of Test Time Adaptation (TTA), aiming to mitigate the impact of covariate shift. Assuming the availability of a limited amount of unlabeled images from test distribution, these approaches try to capture the approximate statistics of the test domain. By adapting the network weights based on these statistics, they show promising results in alleviating the adverse effects of the distribution gap between training and testing data. While TTA has shown considerable potential in the image domain, its effectiveness in the video domain remains a topic of inquiry. Video data introduces additional complexities and temporal dynamics, which may affect the transferability of TTA-based observations from images to videos. As such, further research is needed to investigate optimal ways for applying TTA techniques in the video domain.

In the following, we summarize the research questions investigated in this thesis motivated by the discussions above, followed by our proposed solutions and contributions.

3.2 Research Questions and Contributions

The contributions of this work are divided into three main chapters, looking into different challenges in image and video processing:

3.2.1 Image Domain

Question 1: How robust is an image classifier to background clutter? How can we mitigate the performance drop caused due to this effect? Our study focuses on a scenario where the input image exhibits severe background clutter, and the object is located at a random position in the image. We experimentally observe that these data characteristics

considerably deteriorate the classifier's performance. To facilitate this issue, we propose a novel approach that geometrically transforms the input image to discard the background clutter and center the main content, thereby enhancing classification accuracy. To this end, We train an additional network to acquire the desired transformations for gradually enhancing the classifier's input. Accordingly, we name the proposed model *Sequential Spatial Transformer Network (SSTN)* as it sequentially transforms the input image intending to remove the background noise. This research question is discussed in **Section 5.3**.

Question 2: Can the transformations generated by SSTN be leveraged for curriculum learning? Curriculum learning is a widely recognized technique for improving the training of neural networks in terms of training stability and final accuracy. It involves starting the training process with easier tasks and gradually increasing the difficulty level. However, automatically generating the curriculum remains a challenge. Inspired by the progressive image transformations performed by SSTN, we hypothesize that these transformations create a spectrum of data ranging from easy to hard. Hence, we explore multiple ways to utilize the SSTN-generated intermediate data during the training in the context of curriculum learning. We empirically demonstrate that the proposed training scheme considerably enhances classification performance. This research question is discussed in **Section 5.4**.

Question 3: Can the principles of SSTN be extended to multiple objects? In our study of SSTN, we focused on a simplified scenario where images contained artificial clutter, with only one actual object present in the scene. Building on the insights gained from SSTN, we explore the possibility of extending this approach to a more challenging scenario with multiple actual objects. In this case, the model is expected to identify and focus on the *salient* object. This outline is inspired by human perception, where in the first glance, the attention is automatically drawn to the most salient object in the scene [BSI13]. However, the task of identifying the salient object is challenging, primarily because of the inherent ambiguity in the definition of saliency. To tackle this, we hypothesize that object size correlates with saliency and consider the largest object in the scene as the most salient. Consequently, we develop a variant of the SSTN model that learns to prioritize the largest/salient object, thereby improving the classification accuracy for that object. Our investigation encompasses several challenges associated with this study, such as the assumption of object size determining saliency and the implications faced due to the unavailability of a specialized dataset for this task. This research question is discussed in **Section 5.5**.

3.2.2 Video Domain

Question 4: How can the performance of video object segmentation models be improved for scenarios involving smaller objects? The subsequent section of the thesis delves into video processing applications, with a specific focus on video object segmentation and tracking. Given the inherent temporal richness of video data, it is natural to employ architectures based on recurrent neural networks (RNNs) to leverage spatiotemporal features. Motivated by this notion, we conduct a thorough investigation of a well-established RNN-based approach for video object segmentation. Through this exploration, we identify limitations of the model when faced with challenging scenarios, particularly when tracking smaller objects and accurately segmenting object boundaries. To address these limitations, we propose two effective solutions based on architecture and training objective enhancements that significantly improve the model's performance in such scenarios. These solutions contribute to improved tracking of smaller objects and more precise segmentation of object boundaries. This research question is discussed in **Section 6.3**.

Question 5: Can we extend the VOS model's performance to effectively process longer videos depicting partially occluded objects? A limitation of RNN-based solutions for processing sequences is the phenomenon known as catastrophic forgetting, whereby the model tends to forget information from earlier time steps as the sequence length increases. Consequently, the performance of such models deteriorates, particularly in occluded scenes and longer videos. To mitigate this effect, we propose a hybrid architecture that combines the strengths of RNNs in leveraging spatiotemporal features with correspondence-matching techniques for improved handling of longer videos. Furthermore, we explore two architecture variations: multi-task learning with optical flow integration and a bidirectional architecture. This research question is discussed in **Section 6.4** and **Section 6.5**.

Question 6: Can we still attain a robust performance in crowded scenes under limited label availability for (various) spatiotemporal tasks? Data labeling is a costly process that poses limitations on unlocking the full potential of data-driven solutions. This scarcity of labeled data is particularly evident in the video domain, where frame-level labels are required. To address this challenge, recent efforts have emerged in the form of self-supervised learning methods aiming to train models for spatiotemporal correspondence learning. However, these methods often rely on color information as a supervisory signal, and their performance significantly degrades in crowded scenes with multiple visually similar objects. To this end, we propose a straightforward yet highly effective framework based on pseudo-labels that directly trains the model on the final objective of correspondence learning. Importantly, our approach eliminates the need for video-level labeled data, allowing for improved scalability and versatility in training data acquisition. This research question is discussed in **Section 7.3**.

Question 7: Can we overcome distributional shifts between training and testing phases for spatiotemporal correspondence learning? As the last contribution of this thesis, we delve into the crucial aspect of model generalization, which greatly influences the practicality and applicability of the learning-based methods. To address the challenges faced due to the covariate shift between training and testing data, we explore the emerging field of Test Time Adaptation, which aims to adapt deep learning models to new distributional settings. Recognizing the abundant structural and temporal information inherent in video data, we investigate several approaches to utilize unlabeled videos for adapting the model to various domain shift scenarios and experimentally confirm the efficacy of these methods for out-of-domain generalization in spatiotemporal correspondence learning. This research question is discussed in **Section 7.4**.

4 Background and Toolbox

In this chapter, we provide an introduction to the background and tools employed throughout this thesis. Firstly, we introduce the three fundamental learning paradigms governing this research: supervised, self-supervised, and reinforcement learning. Subsequently, we provide an overview of the utilized low-level neural network components. This is followed by an explanation of the key neural network architectures and optimization techniques that have enabled the progress of this thesis.

4.1 Learning Paradigms

Deep learning methods can be divided into multiple classes based on the learning objective and the utilization of the labeled data, such as supervised learning, unsupervised learning, semi-supervised learning, meta-learning, and reinforcement learning. The following subsections briefly describe the three main learning categories employed in this thesis.

4.1.1 Supervised Learning

In supervised learning, the model receives direct supervision from labeled data, commonly known as ground truth. It was in the domain of supervised learning that initial breakthroughs in deep learning were achieved (Deng et al., 2009; Krizhevsky et al., 2017). In this learning paradigm, the learner (neural network) is trained to estimate a complex function that maps the data from the input domain to corresponding output labels. This is accomplished by minimizing a task-specific loss function through the use of various optimization algorithms. The objective is to train the model such that it can generalize well, making accurate predictions on unseen data with a distribution similar to the training dataset.

One of the main advantages of supervised learning is stable training and achieving high accuracy when trained on large datasets. However, there is a practical desire to develop algorithms that are less dependent on labeled data, as acquiring such data can be resource-intensive. This motivates the exploration of other learning paradigms, such as unsupervised learning and semi-supervised learning, which aim to reduce reliance on labeled data and provide the potential for more efficient and scalable learning algorithms.

4.1.2 Self-supervised Learning

In contrast to supervised learning, unsupervised methods aim to learn the underlying data distributions without any labeled data [Sch+21]. Self-supervised learning is a sub-category of the unsupervised methods where the model is trained with a supervision signal obtained from the data instead of using human annotations. These approaches have become highly popular in recent years due to their scalability and practical advantages. The goal of these algorithms is mainly to learn data representations that can be transferred to downstream tasks, such as classifications, by designing a variety of pretext tasks with auto-generated labels. Through training the model for solving these tasks, it can extract meaningful representations by learning the visual patterns and the contextual relations in an image. Examples of these tasks are predicting certain properties or relationships within the input data, such as predicting missing parts of an image, the next frame in a video sequence, or the context of a word in a sentence.

Other types of self-supervised techniques include contrastive learning and pseudo-labeling. In contrastive learning, the model is trained to compute features from the input data such that features from members of the same category are pulled together while pushed away from the other classes. Differently, in pseudo-labeling methods, various resources such as prior knowledge or trained models are used to generate the (potentially noisy) ground truth labels.

4.1.3 Reinforcement Learning

Reinforcement learning (RL) is the process of learning by experience. In this methodology, a neural network called *agent* is trained to interact with a defined environment and draw actions from a specific action set to maximize a cumulative reward signal. Hence, the agent receives feedback in the form of positive or negative rewards based on the selected actions, and its goal is to learn an action-selection policy that maximizes the cumulative reward over time. The reinforcement learning process can be summarized as follows:

- The agent observes the current state of the environment.
- The agent selects an action based on the observed state.
- The environment transitions to a new state and sends a reward signal to the agent.
- The agent updates its action policy based on the received signal.

This process is repeated for several steps, known as an episode, to maximize the cumulative reward over time.

One of the main advantages of reinforcement algorithms is their ability to optimize for non-differentiable objectives. As a result, they offer more flexibility in terms of the training objectives compared to other learning categories that are bound to differentiability, opening up possibilities for tackling a broader range of complex and challenging problems. However, training reinforcement learning agents can also be challenging, as they often require large amounts of trial and error to learn effective policies.

4.2 Deep Neural Networks

A deep neural network architecture is often comprised of various layers that are specifically designed to capture the necessary functional requirements for the network to fulfill the task at hand. In the following, we provide an overview of the layers and components used in this thesis.

4.2.1 Fully-connected Layer

A fully-connected layer in a neural network that performs a linear transformation of the input data through matrix multiplication, mapping the input to the output using the equation:

$$y = Wx + b \tag{4.1}$$

Here, $x \in R^{d_{\text{in}}}$ represents the input, $W \in R^{d_{\text{out}} \times d_{\text{in}}}$ denotes the learnable weight matrix, and $b \in \mathbb{R}^{d_{\text{out}}}$ is the learnable bias vector. The dimensions d_{in} and d_{out} correspond to the input and output dimensions, respectively. The term fully-connected layer originates from the concept that every neuron or node in a given layer is connected to every neuron in the subsequent layer through the dense matrix of W; hence, this layer is also commonly referred to as *dense layer*.

Due to the linear nature of the operations performed within this layer, it inherently lacks the capability to effectively model complex non-linear functions. Therefore, employing multiple layers in conjunction with non-linear activation functions is necessary to approximate non-linear functions.

4.2.2 Convolution Layer

Convolutional networks share similarities with fully-connected layers in terms of their learnable weights and biases. However, convolutional layers introduce a distinct mechanism by applying trainable filters/kernels to different regions of the input image in a sliding-window manner, performing dot products. In the case of images, the 2D convolution operation can be formalized as [GBC16]:

$$(I * K)(i, j) = \sum_m \sum_n I(m, n)K(i - m, j - n) \tag{4.2}$$

Here, I and K represent the image and kernel, while (i, j) are locations within the structured input grid (pixels in the case of an image). Similar to fully-connected layers, convolutions are usually combined with non-linear activation functions to enhance the expressive power of the network.

Convolutional layers are widely used in deep learning models and have contributed significantly to advancements in computer vision. They excel at capturing visual patterns and extracting powerful features from visual inputs by leveraging several inductive biases to exploit specific characteristics of visual data [Li20; GBC16]. These inductive biases are briefly described in the following.

Parameter sharing: Convolutional layers leverage the assumption that visual patterns, such as edges and blobs, are recurring patterns across an image. As a result, they perform parameter sharing by applying the same kernels to different image regions, eliminating the need to assign new weights for every part of the image. This strategy reduces the number of parameters in the network and enables the network to generalize better and effectively handle variations of the same pattern across different regions of the input image.

Local connectivity: Convolutional layers usually employ small-size kernels and exploit the locality of connections. This is particularly advantageous when processing high-dimensional input data like images and videos, as it significantly reduces the computation costs compared to connecting each element in the input and output. This results in efficiently capturing spatial dependencies within the input, focusing on smaller receptive fields rather than considering the entire input at once.

Regular input structure: Visual inputs exhibit a structured arrangement within a fixed and regular grid, such as a spatial image grid or a spatiotemporal video grid. This characteristic distinguishes them from irregular data formats like point clouds. The regularity of the input domain offers a distinct advantage to convolutional networks, as they can process data using kernels of fixed shapes, facilitating efficient computations.

4.2.3 Activation Layer

In deep learning architectures, various linear operations, such as convolutions, are interconnected using non-linearity or activation layers. This enables the network to model highly complex and nonlinear functions. One of the most commonly used activation functions is the Rectified Linear Unit (ReLU), defined as:

$$\mathrm{ReLU}(x) = \max(0, x) \tag{4.3}$$

Other popular activation functions include the Sigmoid function:

$$\mathrm{Sigmoid}(x) = \frac{1}{1 + \exp(-x)} \tag{4.4}$$

and the Hyperbolic Tangent (Tanh) function:

$$\mathrm{Tanh}(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \tag{4.5}$$

In today's deep learning architectures, the ReLU activation function is favorable against Sigmoid and Tanh functions for several practical reasons, such as improved computational efficiency, faster convergence, and avoidance of saturation issues faced in Sigmoid and Tanh non-linearities. However, these activation functions still serve specific purposes and have their own applications. Sigmoid is frequently utilized in the output layer for binary classification problems, where the goal is to obtain a probability value between 0 and 1. Tanh, on the other hand, proves useful in scenarios where the output needs to be normalized within the range of -1 to 1, particularly when dealing with data centered around zero.

Another commonly used activation function used as the output layer is the softmax layer which maps the output of the neural network to a multi-class probability distribution:

$$\mathrm{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{4.6}$$

Utilization of different activation functions gives deep learning models more expressive power and enables them to model complex functions, capturing various patterns and relationships in the input data.

4.2.4 Pooling and Un-pooling Layers

Pooling layers are frequently used in conjunction with convolution layers to reduce the feature's spatial dimensions. Dimension reduction serves practical purposes, such as reducing

memory consumption, capturing the most salient features of an image, and providing spatial invariance by lowering sensitivity to small spatial shifts in the input image. The downstream task can also dictate downsampling; for example, in image classification, it is necessary to condense the image information into a feature vector that can be mapped to a classification label. This can be achieved by employing different kinds of pooling functions.

In the pooling operation, the input features are divided into windows of size $k \times k$, where the windows are spaced by the *stride* parameter. The output feature map is then computed by applying the function f to the content of each kernel window in the input. Depending on the pooling variant, f can serve various purposes, such as computing the average or maximum value in a window. Considering a simplified scenario with no padding and dilation, the output feature dimensions (height, width) can be computed as follows:

$$d_{out} = \lfloor \frac{d_{in} - k}{s} + 1 \rfloor, \tag{4.7}$$

where d stands for dimension, k is the kernel size and s is the stride parameter.

In structured prediction tasks such as segmentation, where a label is needed for every pixel in the image, generating an output label map at the original input resolution is required. To achieve this, after encoding the input image through multiple convolutions and pooling layers, several un-pooling or upsampling layers are applied to decode the features into the output space and restore the original resolution. Upsampling layers increase the spatial size of the features and can be implemented using various interpolation techniques, such as *nearest neighbor* or *bilinear* interpolation. The architecture resulting from applying a series of pooling and upsampling layers is often referred to as an encoder-decoder architecture. This design is further discussed in Section 4.3.2.

4.2.5 Recurrent Neural Networks (RNNs)

Feed-forward neural networks consisting of fully-connected layers and convolutions do not have the tools for modeling memory functionalities for incorporating past information. This is crucial for sequence processing tasks such as machine translation and video processing, where the model needs to consider the temporal context. RNNs enable the model to keep track of past events via a feedback connection that writes to a memory vector called hidden state h_t.

$$\begin{aligned} h_t &= f_{wx}(h_{t-1}, x_t) \\ y_t &= f_{wy}(h_t) \end{aligned} \tag{4.8}$$

where f_{wx} and f_{wy} are small neural networks with appropriate activation functions selected to facilitate the model's training.

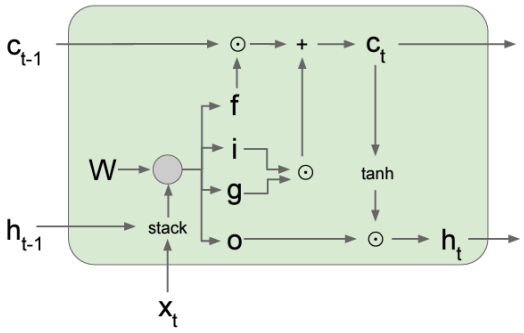


Fig. 4.1: The overall LSTM architecture. Image taken from [Li20].

The vanilla RNNs described above suffer from several limitations, such as unstable training due to the well-known vanishing gradient problem caused by the chain multiplication of the gradients over time. Furthermore, the limited capacity in the memory vector h_t makes it difficult to remember long sequences, leading to the model losing information about the earlier time steps and only keeping the short-term memory. In the following, we discuss architectural designs that address these limitations.

Long-Short Term Memory (LSTM). To address the challenges in training the RNNs, Hoch *et al.* propose a modified architecture that significantly improves the stability in training the vanilla RNNs [HS97]. To this end, they propose a recurrent model with several gating mechanisms as well as an additional *cell state* that facilitates the gradient flow during the optimization process by providing shortcut paths for passing the gradients to earlier time steps and resolving the vanishing gradient problem. The equations determining the operations in LSTM can be summarized as follows [Li20]:

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \text{Tanh} \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \tag{4.9}$$

$$\begin{aligned} c_t &= f \odot c_{t-1} + i \odot g \\ h_t &= o \odot \text{Tanh}(c_t) \end{aligned} \tag{4.10}$$

where i, f, o, g are the introduced gates, W is the weight matrix of a fully-connected layer, σ represents the sigmoid functions, and c_t is the cell state. The overall architecture of the LSTM module is shown in Figure 4.1.

Convolutional LSTM Layer (ConvLSTM). In the original LSTM architecture [HS97], the recurrent layers are implemented using fully-connected layers followed by sigmoid and Tanh nonlinearities. However, this architecture may not be suitable for processing high-

dimensional visual data. To address this limitation, Xingjian *et al.* propose an LSTM variant that replaces the fully-connected layers with convolutions [Xin+15]. This modification allows the LSTM to better leverage the power of convolutional operations for capturing spatial patterns in visual data. The resulting architecture, known as Convolutional LSTM, is governed by the following set of equations [Xin+15]:

$$\begin{aligned} i_t &= \sigma(W_{xi} * x_t + W_{hi} * h_{t-1} + W_{ci} \circ c_{t-1} + b_i) \\ f_t &= \sigma(W_{xf} * x_t + W_{hf} * h_{t-1} + W_{cf} \circ c_{t-1} + b_f) \\ c_t &= f_t \circ c_{t-1} + i_t \circ \tanh(W_{xc} * x_t + W_{hc} * h_{t-1} + b_c) \\ o_t &= \sigma(W_{xo} * x_t + W_{ho} * h_{t-1} + W_{co} \circ c_t + b_o) \\ h_t &= o_t \circ \tanh(c_t) \end{aligned} \tag{4.11}$$

Variants of ConvLSTM layers are widely utilized for video processing applications such as action recognition, video segmentation, and tracking.

4.2.6 Attention Layer

In [Bah+16], Bahadanau et al. introduced the attention mechanism as a solution to address the limited memory issue in LSTMs. The main concept is to incorporate an additional context vector to augment the memory h_t by providing contextual information related to the current time step. In [Bah+16], the authors developed the attention layer for machine translation tasks, where the goal is to translate an input sentence from one language to another. At the time, this task was approached using an RNN-based architecture, where the model first encoded the sentence into a single vector and then gradually decoded it into the target language. The formulation was as follows:

$$h_i = f(h_{i-1}, y_{i-1}) \tag{4.12}$$

Here, h_i represents the hidden state in LSTM, y_i is the model output (decoded word) at time step i, and f denotes a neural network. To improve this formulation, Bahadanau et al. proposed the following modification:

$$h_i = f(h_{i-1}, y_{i-1}, c_i) \tag{4.13}$$

with c_i representing the context vector computed for time step i. The context vector scans all the input words from the previous and potentially future words and summarizes them into a single vector based on their similarity to h_i. Mathematically, this can be expressed as:

$$c_i = \sum_{j}^{T_x} \alpha_{ij} s_j \tag{4.14}$$

Here, α_{ij} represents the attention weights. In the context of machine translation discussed in [Bah+16], s is the mapping from the input sentence to a sequence of embeddings through the encoder network. The normalized attention weights are then computed as follows:

$$\begin{aligned} \alpha_{ij} &= \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})} \\ e_{ij} &= a(h_{i-1}, s_j) \end{aligned} \tag{4.15}$$

where a is a simple neural network such as an MLP that computes the un-normalized attention weights. Although the attention layer was initially developed for machine translation, it has been successfully applied to various domains, with slight modifications in the definition of the context vector to adapt to specific tasks.

Following the tremendous success of attention mechanisms in enhancing the performance of various sequential processing tasks, this module has been extensively incorporated into numerous architectures. More recently, attention has even been utilized as a standalone component, eliminating the need for the LSTM layer. In this design, a key-query operation is employed to retrieve the output based on attention weights calculated from the similarity between a query and the inputs. Formally, given inputs $x \in R^{N \times D}$ and a query $q \in R^D$, the output is computed as follows:

$$\begin{aligned} k = xW_k,\ v = xW_v \\ e_{ij} = \frac{q_j \cdot k_i}{\sqrt{D}},\ a_{ij} = \frac{\exp(e_{ij})}{\sum_i \exp(e_{ij})} \\ y_j = \sum_i a_{ij} \cdot v_i \end{aligned} \tag{4.16}$$

Through these operations, the input x is transformed into a set of key (k) and value (v) vectors using fully-connected layers with weight matrices W_k and W_v, respectively. The similarity between the query q and the keys k is calculated in terms of the dot product, normalized by the square root of the dimensionality D. The attention weights (a_{ij}) are then obtained by applying the softmax function to the similarity scores (e_{ij}). Finally, the output (y_j) is computed as the weighted sum of the values (v_i) using the attention weights (a_{ij}).

Axial attention. The attention layer, commonly used in deep learning models, has high memory consumption of $O(N^d)$, with d representing the number of data dimensions. For instance, in the case of images, $d = 2$ (width and height), and for video data, $d = 3$ (width, height, and time).

This high memory consumption becomes a significant obstacle when applying the attention layer to high-dimensional data. To overcome this limitation, a variant called axial attention was proposed in [Ho+19]. The axial attention layer breaks down the attention operation

into d hops or passes, allowing attention to be performed across each dimension separately. This division across dimensions reduces the memory requirements to $O(dN)$, which is a substantial improvement compared to the original attention layer. Essentially, the axial attention layer performs the attention operation independently across each dimension and aggregates the results. This design enabled applying attention to high-dimensional data and designing architectures that are entirely based on attention mechanism, such as AxialNet in [Wan+20b].

4.2.7 Regularization Layers

There are several techniques in the literature for enhancing the training process and improving the generalization of neural networks by mitigating the effects of overfitting. These methods, known as regularization, introduce additional constraints to the loss function or modifications to the model during training. They aim to prevent the model from overly focusing on the training data and encourage it to learn more generalized patterns that can be applied to unseen examples. In the following, we overview three main regularization strategies used in this thesis.

Weight regularization. Weight regularization, also known as weight decay, is a simple yet effective technique used in neural networks to facilitate overfitting by adding an additional constraint on the L_1 or L_2 norm of the network weights [KH91]. Overfitting occurs when a model obtains high accuracy on the training set but has a low performance on the unseen test data. One contributing factor to overfitting is the presence of large network weights, which, according to [RM99], can adversely affect generalization by causing significant output changes for even small input variations. This effect can lead to the network becoming too specialized to the training data, hindering its ability to make accurate predictions on unseen test data. Weight regularization counteracts these issues by simply imposing penalties on the magnitude of the weights during the training process. This is implemented by adding a regularization term proportional to the magnitude of the weights to the loss function.

Two commonly used weight regularization techniques are L_1 and L_2 regularization. In these variants, the terms $\gamma||W||_1$ and $\gamma||W||_2$ are added to the training objective, respectively. In this setup, γ determines the strength of the regularization, and W represents the network weights.

Feature normalization. Normalization layers are components developed to normalize the intermediate features in a neural network to improve stability and convergence during training. There are different normalization methods based on the way that the normalization parameters are computed. *Batch normalization* is a commonly used layer paired with convolution layers, normalizing the features by adjusting them based on the mean and

standard deviation acquired from the input batch [IS15]. The operation performed by this layer can be summarized as follows:

$$y = \frac{x - \mathrm{E}(x)}{\sqrt{\mathrm{Var}(x) + \epsilon}}\gamma + \beta \tag{4.17}$$

where γ and β are additional learnable parameters that allow the network to adjust the features more flexibly. By normalizing the inputs, batch normalization helps address the issue of internal covariate shift, where the input data distribution to each network layer may change during training. The behavior of the batch normalization layer differs between the training and testing phases. During training, the mean and standard deviation values are computed from the current batch, which allows the model to adapt to the statistics of the specific batch. Training with larger batch sizes is generally more stable and can better represent the overall data statistics and characteristics. During inference, however, the model must be independent of the batch size. Therefore, the accumulated parameters from the training phase, including the running mean and standard deviation, are utilized during inference. These statistics capture the overall distribution of the training data.

However, the effectiveness of models utilizing batch normalization heavily relies on training with an appropriate batch size, which can be problematic when working with large models or high-dimensional inputs that may not fit into memory with a large batch. To tackle this challenge, several alternative normalization layers have been proposed in the literature, including *layer normalization* and *group normalization*. These normalization techniques compute the normalization parameters across different dimensions rather than relying on the batch size alone, thereby mitigating the dependency on batch size and enabling more flexible training setups.

Dropout. One of the challenges in training neural networks is that only a small subset of neurons activate during the training process. Consequently, the model tends to focus on specific characteristics of the data and disregards others. This selective attention can amplify overfitting and hinder generalization. To address this issue, the dropout layer was introduced in [Hin+12]. During training, this layer randomly drops a portion of the neurons, encouraging the model to consider all aspects of the data and train all network weights, promoting better generalization. The Dropout layer helps mitigate overfitting and enhances the robustness of neural networks by preventing the model from relying too heavily on a subset of neurons. We note that the dropout layer is only used during training and deactivated during inference.

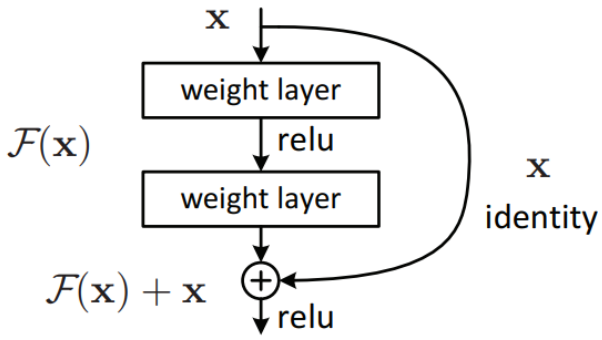


Fig. 4.2: The architecture of ResNet layer. Image taken from [He+16].

4.3 Deep Learning Architectures

Designing neural network architectures that effectively combine the low-level layers to model the requirements for optimal processing of input data, leading to improved accuracy, stable training, generalization, and efficiency, has been an active research area since the advent of deep learning [He+16; Sze+17; SZ14]. Over the past decade, remarkable progress has been made in this domain. In the following, we will briefly describe three key architectures used in this thesis. These designs have proven influential in solving different problems and gained prominence in the vision community, inspiring the development of various architectures leading to further progress in the field.

4.3.1 ResNet

Despite the success of several early deep neural network architectures in image classification [KSH12; SZ14], designing and customizing layers to facilitate successful training and improving the model accuracy were challenging. While it has been evident that increasing the depth of neural networks enhances their representative capacity, it also introduced challenges in training due to the vanishing gradient problem. To address this limitation, He *et al.* proposed Residual Networks, ResNet in short [He+16]. In ResNet, a modular design of convolution layers was combined with a residual connection that allows information to bypass certain layers and directly flow through the network (see Figure 4.2). By incorporating these skip connections, ResNet effectively mitigates the vanishing gradient problem and facilitates the training of deeper models. The idea of residual connections has been broadly adopted in numerous recent architectures and has become a common feature in deep learning frameworks [Dos+20].

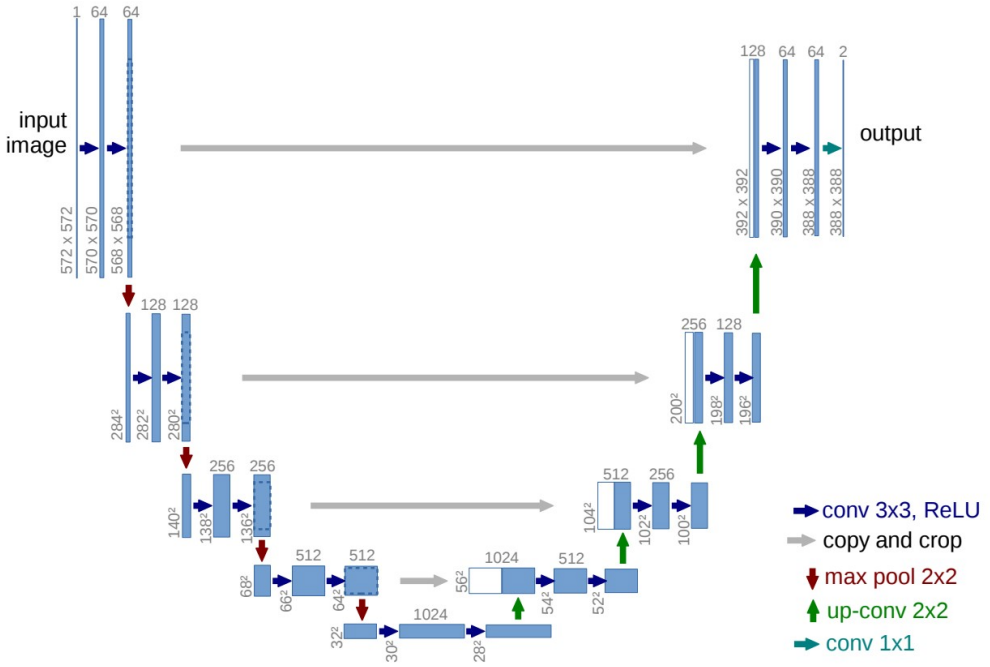


Fig. 4.3: The overall UNet architecture. Image taken from [RFB15].

4.3.2 UNet

As mentioned in Section 4.2.4, the encoder-decoder architecture is a widely used framework in deep learning. The encoder component progressively decreases the spatial dimensions and increases the number of channels while the decoder performs the inverse operation, decoding the computed features to generate the desired output. However, a drawback of this architecture is the loss of fine details caused by extensive down-sampling. This limitation becomes problematic in tasks like segmentation, where accurate preservation of fine details and small objects is essential. To address this issue, the UNet architecture proposed incorporating skip connections between the encoder and decoder layers [RFB15], as shown in Figure 4.3. These skip connections serve as direct pathways between the encoder and decoder sections at different resolutions. By preserving and integrating features from higher resolution levels, UNet enables the decoder to access and leverage fine-grained information that would be lost otherwise. This mechanism significantly improves the model's ability to capture and segment fine details and small objects.

4.3.3 Feature Pyramid Networks (FPN)

The FPN (Feature Pyramid Network) architecture was initially proposed to tackle the challenge of multi-scale object detection and segmentation by processing visual input at

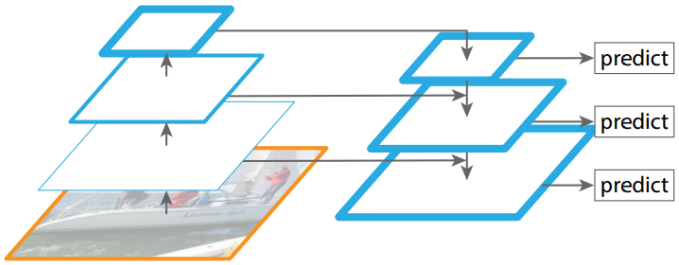


Fig. 4.4: Feature pyramid network architecture. Image taken from [Lin+17].

different scales [Lin+17]. The concept of multi-scale processing has been successfully employed in traditional computer vision methods, such as computing SIFT features [Low04], to achieve scale invariance. To bring the advantages of multi-scale processing into deep learning architectures, Lin *et al.* introduced FPN, an architecture that generates a feature pyramid from a backbone network, typically a convolutional neural network like ResNet [He+16]. In FPN architecture illustrated in Figure 4.4, the vertical pathway involves upsampling the higher-level feature maps to match the spatial resolution of the lower-level feature maps, allowing the network to propagate high-level semantic information to finer spatial scales. Moreover, horizontal connections are established between the upsampled higher-level feature maps and the corresponding lower-level feature maps, fusing the high-level semantic information with the more detailed spatial information in the lower-level feature maps. This utilization of information across multiple scales enables the network to detect objects of varying sizes while maintaining precise localization.

4.4 Optimization

Backward propagation of errors (backpropagation) is currently the dominant algorithm for training deep-learning models. After computing the loss function as an error measurement between the model predictions and the ground-truth labels, the gradients of the error with respect to the network's parameters are computed using the backpropagation algorithm [GBC16]. Having the gradients, various optimization methods can be employed to minimize the loss function and train the network parameters.

The origins of backpropagation can be traced back to [Wer74], a Ph.D. thesis written by Paul Werbos in 1974. However, at the time, the significance of his work went largely unnoticed by the neural network community. In 1986, Rumelhart *et al.* published a seminal paper titled "Learning Representations by Backpropagating Errors," reintroducing backpropagation as a promising way for training neural networks. Since then, numerous advancements have been made to enhance the efficiency and stability of gradient descent optimization

algorithms, aiming to establish better ways for updating network parameters to achieve improved convergence. The general form of gradient descent involves updating the network weights to reduce a total cost function J:

$$\theta = \theta - \alpha \nabla J(\theta) \tag{4.18}$$

where θ represents the network weights, α is the learning rate determining the size of the update, and $\nabla J(\theta)$ is the gradients of J with respect to θ computed by the backpropagation algorithm. In practice, parameter updates are performed using a mini-batch of data as gradients from individual data points can be very noisy, and computing updates from the entire dataset is very time-consuming. In the following, we overview two optimization algorithms employed in this thesis.

4.4.1 SGD with Momentum Optimizer

One of the limitations of vanilla SGD in Equation 4.18 is gradient oscillation around local minima. To address this issue, Qian (1999) proposes the use of a momentum parameter that controls this behavior. The updated equations are as follows:

$$m_t = \gamma m_{t-1} + \alpha \nabla_\theta J(\theta) \tag{4.19}$$

$$\theta = \theta - m_t \tag{4.20}$$

If we consider the loss landscape to be ravine-shaped, where the objective is to find parameters that yield the minimum loss within the ravine, the momentum intuitively controls the velocity around the bottom. This is achieved by reducing the momentum term for dimensions where the gradients alternate in sign.

4.4.2 Adam Optimizer

One of the limitations of the optimization algorithms discussed above is using the same learning rate for updating all the network parameters. However, it would be preferable to assign a higher learning rate to the weights associated with less frequent data features. The Adam optimization algorithm [KB14] addresses this issue by incorporating information about accumulated past gradients. Assuming $g_t = \nabla_\theta J(\theta_t)$, Adam computes moving average of g_t and g_t^2 as:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{4.21}$$

$$\nu_t = \beta_2 \nu_{t-1} + (1 - \beta_2) g_t^2 \tag{4.22}$$

where β_1 and β_2 are hyperparameters. Subsequently, the updated equation can be summarized as follows:

$$\theta_{t+1} = \theta_t + \frac{\alpha}{\sqrt{v_t} + \epsilon} m_t \tag{4.23}$$

with α as the learning rate. From Equation 4.23, we can see that the learning rate is adaptively tuned based on past gradients, and a higher learning rate is assigned to parameters with lesser past gradients (v_t). For more details, please refer to [KB14].

5 Cluttered Image Classification

5.1 Introduction

Standard classification architectures are well-known for achieving remarkable performance on specialized image classification datasets containing clean and well-centered images [KH+09; Den+09]. However, their accuracy tends to decline significantly when confronted with images that are cluttered with background noise or lack proper centering [JSZ+15; Hen+21]. While previous research has mainly focused on enhancing network architectures to create more robust classifiers [He+16; Sze+17], this chapter takes a different approach by exploring modifications to the input image itself, aiming to facilitate easier classification for the network. By adopting this orthogonal perspective, we seek automated ways of preprocessing the input data to improve classification accuracy and ultimately enhance the model's performance in the presence of background noise and clutter.

In the first contribution, we design an iterative algorithm that sequential transforms the input image by simple affine transformations and focuses on the main content in the image. We formulate the task as a Markovian Decision Process (MDP) and use Reinforcement Learning (RL) to solve this sequential decision-making problem. We refer to the proposed model as Sequential Spatial Transformer Network, SSTN in short. Thanks to the RL-based formulation in our method, we are not bound to the differentiability of the training objective; hence, we experiment with a variety of differentiable and non-differentiable objectives, such as the cross-entropy classification loss and maximizing the model accuracy.

As our second contribution, we leverage the developed strategy for sequentially modifying the input data toward an easier distribution for curriculum learning. Curriculum learning is a bio-inspired training technique that is widely adopted in machine learning for improved optimization and better training of neural networks regarding the convergence rate or obtained accuracy. The main concept in curriculum learning is to start the training with simpler tasks and gradually increase the level of difficulty. Therefore, a natural question is how to determine or generate these simpler tasks. As SSTNs have been proven to be capable of removing the clutter from the input images and obtaining higher accuracy in image classification tasks, we hypothesize that images processed by SSTNs can be seen as easier tasks and utilized in the interest of curriculum learning. To this end, we study

multiple strategies developed for shaping the training curriculum using the data generated by SSTNs.

In the last section, we extend the proposed SSTN algorithm to the more challenging task of salient object classification in real-world images. To this end, we employ the recent advances in Q-learning for processing high-dimensional input data such as images, as well as a variety of reward function definitions. Unlike the usual setting of having a single object per image in classification datasets, we consider a setup where multiple objects are present in the scene, and the model is expected to distinguish and classify the salient object. We note that the setup is similar to human perception, where our attention is drawn to the most salient object in an image. We perform extensive experiments on the PASCAL VOC dataset and analyze different aspects concerning the method design and training data, such as dataset bias.

The content in this chapter is based on the published papers [Azi+19; Azi+23; Azi+22a].

5.2 Related Work

This section provides the theoretical preliminaries for the RL algorithms utilized in this chapter, followed by an overview of related work in cluttered image classification and curriculum learning fields.

5.2.1 Reinforcement Learning Preliminaries

The goal of reinforcement learning is to learn by experience, similar to the learning process in humans. In this methodology, the learner (often referred to as **agent**) interacts with the environment and is trained to maximize a predefined reward during multiple decision-making or action-selection steps. The theory of modern reinforcement learning algorithms is based on the Markovian Decision Process (MDP) formulation [SB18]. The Markovian property indicates a memoryless process where the future state of a stochastic process can be entirely determined based on the current state without any dependency on the past:

$$P(s_{t+1}|s_0, \ldots, s_t) = P(s_{t+1}|s_t). \tag{5.1}$$

The components describing an MDP are the set of possible states $\boldsymbol{s} \in \boldsymbol{S}$, the set of actions $\boldsymbol{a} \in \boldsymbol{A}$, and the set of rewards $\boldsymbol{r} \in \boldsymbol{R}$. The overall scheme of the MDP is illustrated in Figure 5.1. At each time step t, the agent samples an action a_t from a learned probability distribution. Based on the action taken and interaction with the environment, the agent

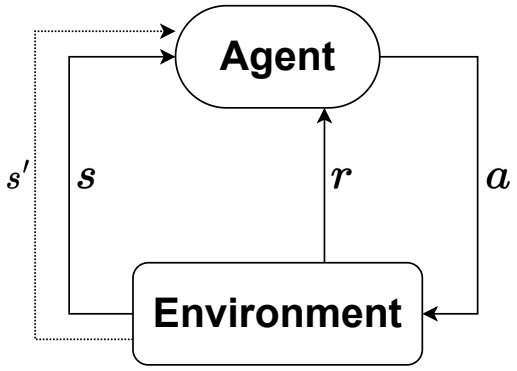


Fig. 5.1: The working of the Markovian Decision Process [SB18]. Starting from state s, the agent selects an action a based on an action selection policy. Following applying the action and interaction with the environment, the agent receives a reward r, and the state of the environment transitions to s'

receives the reward r_t, and the state of the environment changes from s_t to s_{t+1}. The goal of the RL agent is then to maximize the expected overall reward over a sequence of decision-making steps:

$$J(\theta) = E_{\tau \sim p_\theta}\Big[\sum_t r(s_t, a_t)\Big] \tag{5.2}$$

where τ is a sequence of state-action transitions with length of T, referred to as an **episode**. An episode is sampled from the probability distribution based on which the agent selects the actions (p_θ in Equation 5.2). This probability is referred to as the **policy**. Finally, $r = \sum_{t=0}^{t=T} R_t$ is the overall reward obtained at the end of an episode.

The RL literature provides two main categories of algorithms for training the agent: Policy Gradient (PG) and Q-learning. We note that both categories aim to find the optimal policy, which is a state-action probabilistic distribution that results in the highest expected reward. In PG, this is addressed by directly estimating the optimal policy p_θ. In Q-learning, however, a value function $Q_{s,a}$ is learned to estimate the associated value (expected total reward) to each state-action pair. Having this function learned, at each state, the agent greedily selects the action that leads to the maximum reward from each step onward, hence maximizing the overall reward. In the following, we briefly describe both algorithms as they are the foundation for the methods developed in this chapter.

Policy Gradient

As previously mentioned, the main components in RL framework are the State Space (**S**), the Action Space (**A**), and the Reward Signal (**R**). Moreover, an episode τ consists of a sequence of state-action transitions. From Equation 5.2 we have:

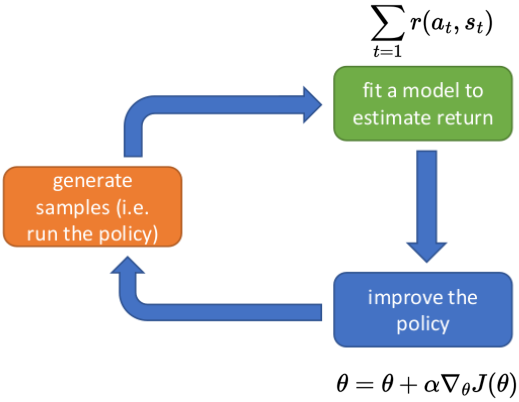


Fig. 5.2: In Reinforce algorithm, the parameters of an action selection policy are learned via backpropagation to maximize the expected returned reward $J(\theta)$. Image Source: [Lev21].

$$J(\theta) = \int p_\theta(\tau) r(\tau) d\tau \approx \frac{1}{N} \sum_i \sum_t r(s_{i,t}, a_{i,t}), \tag{5.3}$$

with t representing time step and i indexing the state-action pair. Assume the policy function p_θ is approximated with a neural network. In Policy Gradient, the parameters θ are directly optimized to maximize the expected reward J:

$$\theta^\star = \text{argmax}(J_\theta). \tag{5.4}$$

Consequently through differentiation of J_θ we have:

$$\nabla_\theta J(\theta) = \int \nabla_\theta p_\theta(\tau) r(\tau) d\tau. \tag{5.5}$$

Using the equality $p_\theta(\tau) \nabla \theta \text{log} p_\theta(\tau) = p_\theta(\tau) \frac{\nabla_\theta p_\theta(\tau)}{p_\theta(\tau)} = \nabla_\theta p_\theta(\tau)$ in the equation above we obtain:

$$\nabla_\theta J(\theta) = \int p_\theta(\tau) \nabla_\theta \text{log} p_\theta r(\tau) d\tau = E_{\tau \sim p_\theta}[\nabla_\theta \text{log} p_\theta(\tau) r(\tau)]. \tag{5.6}$$

Having the gradients $\nabla_\theta J(\theta)$, the parameters are updated with a learning rate of α, as:

$$\theta_{new} = \theta_{old} + \alpha \nabla_\theta J(\theta). \tag{5.7}$$

This is the working mechanism of the **Reinforce** algorithm, as illustrated in Figure 5.2. As can be seen from Equation 5.7, during the iterative optimization in the Rreinforce algorithm, the model weights are tuned towards generating a higher expected reward.

Reinforce algorithm offers a straightforward formulation; however, it suffers from slow convergence and high variance (note that in Equation 5.5, $r(\tau)$ can be noisy, resulting in noisy gradients and slow convergence). A simple idea for reducing the variance is to update

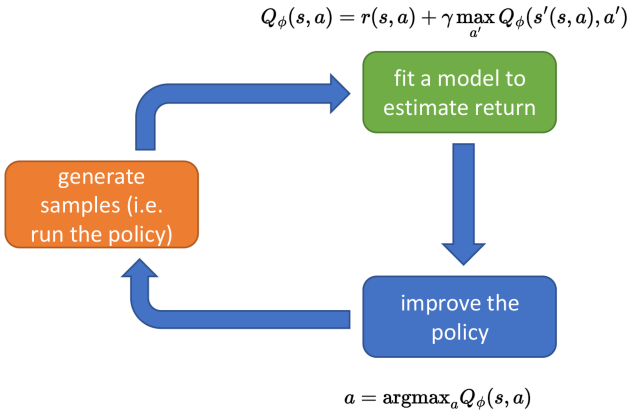


Fig. 5.3: In Q-Learning algorithm, the function Q parametrized by a neural network ϕ is trained to estimate the expected return from the state s for each possible action a. Accordingly, the agent at each state selects an action that maximizes the returned reward. Image source: [Lev21].

Equation 5.3 by replacing $\sum_{t=1}^{t=T} r(s_t, a_t)$ to $\sum_{t=t'}^{t=T} r(s_t, a_t)$, as intuitively, he past rewards subsequent from past actions should not impact the current gradients. This term is called *reward-to-go*. The next solution proposed in the literature is to reduce the variation in $\nabla_\theta J(\theta)$ in Equation 5.5 by subtracting a baseline b:

$$\nabla_\theta J(\theta) = \int \nabla_\theta p_\theta(\tau)(r(\tau) - b)d\tau \tag{5.8}$$

This is the motivation behind Actor-Critic algorithms, as discussed in Section 5.2.1, after introducing the Q-Learning algorithm.

Q-Learning

In Q-learning, the optimal policy is found by estimating the Q function, as shown in Figure 5.3. The Q function estimates the expected total reward from the current state for each possible action as:

$$Q^\pi(s_t, a_t) = E_{\pi_\theta}\left[\sum_{t'=t}^{T}(r(s_{t'}, a_{t'}|s_t, a_t))\right]. \tag{5.9}$$

Having the Q function for each state-action pair, the agent finds the optimal policy by always selecting the action leading to a higher Q value.

The Q function can be approximated with a neural network as a mapping from different possible state-action pairs to the expected return. The objective for training the Q function is the Bellman optimality equation:

$$Q^*_\phi(s,a) = r(s,a) + \gamma \max_{a'} Q^*_\phi(s'(s,a),a'), \tag{5.10}$$

where γ is a hyperparameter and ϕ represents the network parameters. Note that the Equation 5.10 is greedy due to the $\max$ operator. However, at the beginning of the training, when the network weights are initialized randomly, the network output is unreliable. Hence, the greedy action selection strategy in Equation 5.10 may hinder the training and finding the actions that result in higher rewards. To address this issue, an ε-greedy action selection strategy is deployed during the training phase to accommodate a trade-off between the *exploration* of the potential of different state-action pairs and the *exploitation* of the network knowledge. The ε-greedy policy is defined as:

$$a_t = \begin{cases} \text{random action}, & \text{if } p < \varepsilon + (1-\varepsilon) \cdot e^{-\frac{n}{d}} \\ \text{argmax } Q^*(s,a), & \text{otherwise} \end{cases} \tag{5.11}$$

Note that exploration decreases with the progress of training (number of taken steps n with a decay of d). For more details, please refer to [SB18; Lev21].

Double Deep Q-Learning (DDQN). As observed from Equation 5.10, the consecutive updates are highly correlated. This is problematic as estimating the current value and the target Q by the same function (neural network) results in a moving target and slow convergence. This limitation is addressed by utilizing a replay memory [Mni+13] as a way to break the correlation between the training samples. To this end, each experience (s_t, a_t, r_t, s_{t+1}) is stored in replay memory with a large buffer. Accordingly, training samples are randomly drawn from the memory during the training phase. Moreover, [Mni+13] proposes using two separate networks to model the current and the target Q functions to facilitate the moving target problem (hence named Double Q-learning). The target network, however, is not trained with gradient descent, but its weights are periodically updated from the Q network after each N training steps.

Actor-Critic

The Actor-Critic method aims to alleviate the high variance problem in policy gradient by subtracting a baseline from the reward function. This is similar to mean subtraction, a common practice in data normalization for stable training. Specifically, the baseline in the Actor-Critic algorithm is defined as the expected value of the Q function:

$$V^\pi(s_t) = E_{a_t \sim \pi_\theta} \left[Q^\pi(s_t, a_t) \right] \tag{5.12}$$

Essentially, V is the *value* associated with state s_t, computed as the average of outcomes for different actions. The value function V can also be learned by training a neural network to approximate this function. Accordingly, the value network is trained to estimate the total reward from time step t onward. Therefore, the sum of rewards from t to the end of the episode ($\sum_{t=t'}^{T} r(t)$) can serve as the ground truth for training the value network using a proper loss function. For more details, please refer to [SB18; Lev21].

5.2.2 Cluttered Image Classification

Invariance against different transformations is crucial in many tasks, such as image classification and object detection. Therefore, computing invariant features has been a long-standing area of research for decades [Low87; Ste01; YC99; Low99; Mik02; BL02]. In [HS+88], the authors consider corners as representative rotation-invariant features and develop a corner-detection algorithm utilizing intensity information based on the observation that in corners, there should be a high shift in intensity in every direction. Traditional feature extraction methods such as SIFT [Low99; Low04] and SURF [Bay+08], introduce scale invariance into the feature computation by processing the image at several scales. In [Cal+10; RD06; Rub+11], the authors propose more efficient feature extraction alternatives to SIFT by employing techniques such as PCA compression. [Alh+08] improves SIFT by dividing and processing features into multiple frequency domains, resulting in improved feature matching, and Brown *et al.* propose interest-point (intensity local minima and maxima) grouping as a solution for obtaining invariant image descriptors [BL02]. Triggs *et al.* employ a local appearance model toward computing invariant features [Tri04] while in [TC00], the authors suggest computing the Fourier transform of the image intensity profiles as invariant to affine transformations. Though complex, these hand-crafted feature extraction schemes lack generalizability to new scenes and scenarios.

During the last decade, there has been a breakthrough in various areas of computer vision mainly caused by the advances in Convolution Networks [LeC+98; KSH12]. Introducing deeper and more complex classification network architectures [He+16; Sze+15; Hua+17] have alleviated the need for hand-designed feature extraction methods, resulting in achieving high accuracy in challenging datasets such as ImageNet [Den+09]. Although CNNs are translation equivariant and, to some extent, robust to small geometric transformations due to the max-pooling layer used in these architectures, their invariance is limited because of the fixed structure of the learned kernels and receptive fields. Learning-based models usually address this limitation by utilizing data augmentation techniques and including enough variation in the training data [SK19; PW17; MG18]. However, the drawback of data augmentation lies in the assumption of including *all* the possible transformations in the training data. To address this limitation, [Su+19; JK17; Dai+17; Jia+16] propose new

convolution blocks where the kernel shape is learned through backpropagation in contrast to the fixed kernel shape in traditional convolution block. The flexibility in filter shape allows for more invariance to geometric transformations. Differently, [Che+17a; Luo+16] adapt multi-scale processing techniques to increase the receptive fields of CNNs, improving scale-invariability in these models. In [Zha19], the authors deploy anti-aliasing methods such as low-pass filtering commonly used in signal processing fields for improving shift in variance CNNs.

Clutter and background noise are another major challenge limiting the performance of deep learning models [Vas+22]. To this end, several works have developed various attention mechanisms that help the model focus on the image's main content [Cao+15; Has+22; Soy22]. For example, Mnih *et al.* design an RNN-based architecture that starts by processing the whole image at low resolution and iteratively constraining the attention span towards focusing on the main content and high-resolution processing of only the image area that contains the primary object, resulting in improved classification accuracy and computational efficiency [MHG+14]. Similarly, [BMK14; SFR14] employs RNN-based attention models for multi-object classification. Different from these methods, [JSZ+15; LL17] propose algorithms for removing clutter and geometric inconsistencies from the input image by learning the parameters of an affine transformation and then processing the enhanced image for classification.

5.2.3 Curriculum Learning

Curriculum learning is a bio-inspired training technique that is widely adopted in machine learning for improved optimization and better training of neural networks regarding the convergence rate or obtained accuracy. The main concept in curriculum learning is to start the training with easy tasks and gradually increase the difficulty level. Several works in the field of cognitive science and animal training have demonstrated that humans and animals learn faster when the concepts are presented in an organized way with gradually increasing levels of difficulty [Pet04; Ski58; Pav10]. In [Ski58], the authors discuss the importance of successive reinforcement and examine the impact of various scheduling methods to accelerate the learning process in the case of birds. Moreover, Kruger *et al.* [KD09] study the effect of the shaping mechanism concerning the arrangement of the presented tasks based on the complexity factor. According to their analysis, training time increases proportionally with task complexity; however, different ways of shaping and scheduling can considerably reduce this time.

Inspired by these findings, [Ben+09] proposes applying the same methodology to machine learning, which at best aims to imitate the learning process of humans. They formulate

curriculum learning as a training policy that favors easier samples at the beginning of the training and gradually increases complexity. They illustrate the effectiveness of this approach in improving generalization, achieving faster convergence, and finding better local minima in language modeling and shape recognition tasks. Zaremba *et al.* [ZS14] demonstrate the efficacy of curriculum learning in training recurrent neural networks for executing simple computer programs. Their scheme is based on a gradual increase in the length and the number of nested loops in the program. To develop a curriculum learning-based training strategy, [HW19] proposes a sorting method to order the tasks based on an increase in hardness, and in [Gra+17; Gra+16], they utilize learning progress signals such as loss value or prediction gain to automatically select a training path for the neural network in order to enhance learning efficiency. Similarly, in [Ben+15], the authors study the use of curriculum learning in sequence prediction. They suggest replacing the ground truth with the model's prediction following a probabilistic scheme. When training is starting, the ground-truth labels are used for the model to predict the target in the next time steps; then, gradually, the ground truth is replaced with the model's prediction, and the task's difficulty is increased. In [Flo+17], the authors use curriculum learning in the context of goal-oriented Reinforcement Learning where an agent is expected to navigate an environment towards a target by reversing the task instead of placing the agent at a random location and training it to find the goal state, they start from this state and progressively increase the distance, and hence, the task complexity. In another work, Wang *et al.* use curriculum learning to address the challenges of training with imbalanced data where the dataset is not following an independent and identical distribution [Wan+19]. They propose a data sampling method that is responsible for sampling balanced samples and a loss scheduler that dynamically balances the weighting scheme between different loss components in their multi-task setup.

Another closely related line of work is the Teacher-Student training approach [HVD15] in which two agents are trained intertwined such that the teacher network conditions certain aspects of the student network's training, such as the optimization objective or task complexity assortment [Mat+19]. [Jia+18] uses this training policy to mitigate challenges faced when training with noisy data and to prevent the network from overfitting on erroneous labels. To this end, they train a mentor network that generates a curriculum in the form of a weighting scheme for each training sample based on its importance and reliability. Subsequently, this weighting scheme is used for training the base classifier (student network), resulting in improved generalization. In [Fan+18], the authors investigate what constitutes a good teacher in the context of training neural networks. They suggest a framework where teacher and student networks interact in a way where the student undergoes the standard training, and the teacher receives a feedback signal from the student to decide about the training data, loss function, and the hypothesis space.

5.3 Learned Transformations for Robust Classification

Unconstrained image classification, where the main image content is cluttered with background noise, is considerably more challenging than curated scenarios with clean images centered around the target object. To this end, obtaining image features that are robust enough to various transformations and the background noise but also sufficiently discriminative to distinguish between different classes is crucial. One of the solutions proposed in the literature is to simplify the classification by transforming the input image [JSZ+15]. Hence, an important question to ask is *"what are the suitable transformations?"*. In [JSZ+15], the authors introduce Spatial Transformer Networks (STN) for learning the geometric transformations that modify the input such that it is *easier* to classify. In this method, the geometric transformations are learned by minimizing the classification loss. Although it is possible to use different transformations, here we focus only on affine transformation.

The main components of an STN are the *localization network*, the *grid generator* and the *sampler module*, as illustrated in Figure 5.4. The localization network takes the input image and generates the affine transformation parameters. The grid generator computes the location of each output pixel in the input image. To warp the input image based on the estimated transformation, each pixel in the output should be computed using a sampling kernel applied to the input image. The sampler uses the grid generator output and the bilinear sampling kernel to generate output pixels from the input image:

$$V_{ij} = \sum_{n}^{H} \sum_{m}^{W} U_{nm} \text{max}(0, 1 - |x_i - m|) \text{max}(0, 1 - |y_j - n|) \, \forall i, j \in [1 \dots H], [1 \dots W], \tag{5.13}$$

where H and W are the height and width of the image respectively, and V and U are the corresponding pixel values in the output and input image. The coordinate (x_i, y_i) is the location in the input where the sampling kernel is applied. The sampling module is differentiable within a local neighborhood:

$$\frac{\delta V_{ij}}{\delta x_i} = U_{nm} \text{max}(0, 1 - |y_j - n|) \begin{cases} 0 & \text{if } |m - x_i| \geq 1, \\ 1 & \text{if } m \geq x_i, \\ -1 & \text{if } m \leq x_i. \end{cases} \tag{5.14}$$

and similarly for $\frac{\delta V_{ij}}{\delta y_j}$. Therefore, the parameters of the localization network are gradually updated using backpropagation through the classification loss within a local window.

In this work, we take a similar approach as in [JSZ+15] to modify the input data domain toward attaining a higher classification accuracy. We formulate the transformation task as a sequential decision-making problem, in which, instead of finding a one-step transformation,

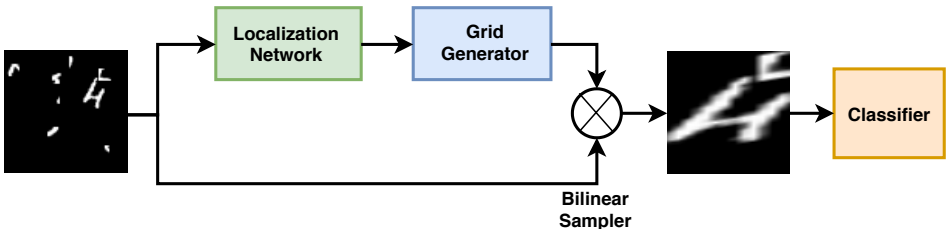


Fig. 5.4: An overview of the components in STN architecture. The localization network generates the parameters of an affine transformation. The grid generator together with the sampler module generates the transformed image.

the model searches for a combination of discrete transformations to improve the performance. We use RL for solving the search problem and apply both Reinforce and Actor-Critic algorithms [Sut+00; Sut84] and experiment with different reward designs, including maximizing classification accuracy and minimizing classification loss. As such, the proposed method is flexible in potentially including both differentiable and non-differentiable types of transformations and training objectives thanks to the RL-based formulation. Unlike [JSZ+15], our method is not limited by partial differentiability and learns an interpretable sequence of transformations, modifying the input toward an easier distribution.

5.3.1 A Reinforcement Learning Approach for Sequential Spatial Transformer Networks

Our goal is to learn a sequence of simple image transformations $T = T_n \cdot T_{n-1} \cdot ... \cdot T_0$, which is applied to the input image and helps the classifier achieve better performance. There are different image adjustments, including geometric transformations and filtering methods. Here, we only consider affine transformations. The hypothesis is that the right affine transformation can act as an attention module by focusing on relevant content in the image. Consequently, this would simplify the classification.

To this end, we decompose the affine transformation into a sequence of specific and discrete transformations instead of applying it in one step as in [JSZ+15]. We formulate the problem of finding the affine parameters as an MDP and aim to learn picking the right transformation at every time step of the sequence. Figure 5.5 shows our proposed architecture.

MDP Framework for SSTN

As mentioned in Section 5.2.1, the main parts in an RL framework are **S**, **A**, and **R**. In this section, we elaborate on these elements in formulating our task.

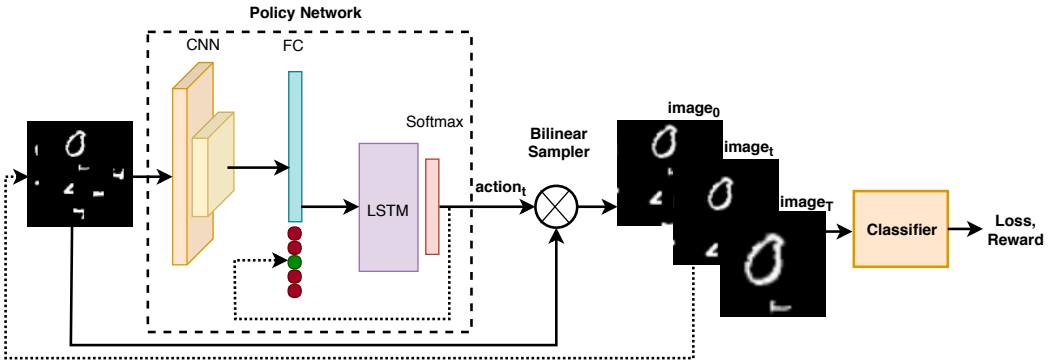


Fig. 5.5: SSTN architecture for finding the sequential affine transformation $T = T_n \cdot T_{n-1} \cdot ... \cdot T_0$. We compare the performance of different policy architectures with and without using LSTM. The last layer of the policy network is a softmax which outputs the probability distribution of the actions. When using LSTM, the one-hot encoded action from the previous time step is merged into the feature map of $image_t$. At each time step, an action corresponding to transformation T_i is sampled from the policy and applied to the image.

State Space (S): We consider two state space definitions and experiment with both of them. First, we define a state as the transformed image at step *t* ($image_t$ in Figure 5.5). Second, we define the state as a combination of the current transformed image ($image_t$) and the previous action (a_{t-1}). We merge the one-hot encoded action from the last time step into the state as $s_t = (image_t, action_{t-1})$. To keep track of the order of sampled actions in different time steps, the model utilizes an LSTM module [HS97], which is a recurrent neural network with gate functions to avoid the vanishing gradient problem. This formulation is closer to the Markovian assumption as the information from past actions helps the network to learn the proper order of applying the transformations.

Action Space (A): Every action $action_t$ is a specific transformation sampled from the policy, which is applied at time step *t* and slightly transforms the image and the state. In order to construct an affine transformation, we define the action space as $A = \{Translation, Rotation, Scale, Identity\}$. The episode length is fixed to *T* for all images, and having the *Identity* transformation allows for stopping the process for individual images before reaching *T*. Having a fixed episode length allows us to train our model in mini-batches.

Reward (R): The agent learns the task while maximizing the reward; therefore, the reward definition has to enfold the objective of the task. An intuitive reward definition would be based on classification accuracy since the goal is achieving higher accuracy. Accordingly, we give a discrete reward of **+1** when a label prediction changes from false to correct as the result of applying an action, and **-1** for the opposite case; other cases get **0** reward:

$$r_1 = \begin{cases} \mathbf{1} & \text{if } (pred_{t-1} \neq label \wedge pred_t = label), \\ \mathbf{-1} & \text{if } (pred_{t-1} = label \wedge pred_t \neq label), \\ \mathbf{0} & \text{otherwise.} \end{cases} \tag{5.15}$$

Here $pred_{t-1}$ and $pred_t$ are predicted labels before and after applying the action at time step t a_t.

Moreover, we can address maximizing the accuracy by minimizing the classification loss similar to [JSZ+15]. This way, the reward design is simply the negated loss:

$$r_2 = -loss. \tag{5.16}$$

In this case, the reward is always negative as loss is a positive value; therefore, the maximum expected reward would be zero. It means that the model tries to learn a policy that pushes the classification loss toward zero.

Additionally, we can consider the reward as the loss difference between consecutive time steps:

$$r_3 = loss_{t-1} - loss_t, \tag{5.17}$$

where t is the time step. With this reward definition, the model tries to maximize the difference in loss values between every two following steps toward a positive reward. In other words, the model tries to pick an action resulting in a smaller loss value than the previous time step.

After defining **S**, **A**, and **R**, we use the algorithms introduced in Section 5.2.1 to learn combining a set of discrete transformations for improving the classifier performance. First, we use the PG algorithm mainly due to its effectiveness and simplicity. In the next step, we extend our implementation to the Actor-Critic algorithm. More details about the Actor-Critic training algorithm for one epoch are presented in algorithm 1 (PG algorithm has a similar pseudo-code without training the additional value network).

5.3.2 Experimental Setup

In this section, we discuss the experimental setting including the datasets, implementation details, experimental results, and ablation studies.

Algorithm 1: Actor-Critic Training Algorithm for SSTN (one epoch).

Input : Images and classification labels

Output : Classifier and sequential spatial transformer policy

Initialize classifier (ψ)*, actor* (θ)*, and critic* (ϕ) *networks*

for *Image = 1 : N* **do**

  $initialize\ state\ \boldsymbol{s},\ t = 0,\ RewardValueList = [empty]$

  while $t <$ *episode-length* **do**

    $p_\theta(a_t|s_t) = \boldsymbol{actor}(s_t)$

    $sample\ the\ action:\ a_t \sim p_\theta(a_t|s_t)$

    $image_{t+1} = \boldsymbol{apply\_action}(image_t, a_t)$

    $\boldsymbol{s_{t+1}} = (image_{t+1}, \boldsymbol{one\_hot}(a_t))$

    $predictions_t = \boldsymbol{classify}(image_t, labels)$

    $reward_t = \boldsymbol{compute\_reward}(predictions_t)$

    $value_t = \boldsymbol{critic}(s_t)$

    $append\ (reward_t, value_t)\ to\ RewardValueList$

    $t\ +=\ 1$

  end

  Update parameters:

  $\Delta\phi = \frac{d}{d\phi}\boldsymbol{MSE\_loss}(v_t, \sum_{t=t'}^{t=T}\gamma^{T-t}r_t)$

  $\Delta\theta = \frac{d}{d\theta}\boldsymbol{E}[log p_\theta(\sum_{t=t'}^{t=T}\gamma^{T-t}r_t - v_t)]$

  $\Delta\psi = \frac{d}{d\psi}\boldsymbol{CrossEntropy\_loss}(predicted\_labels, labels)$

  $\boldsymbol{Empty}(RewardValueList)$

end

Datasets

We evaluate our method using cluttered MNIST [MHG+14] and cluttered Fashion-MNIST datasets. Both datasets consist of 10 classes, encompassing digits in the former and clothing items in the latter. These datasets have been used by several works to demonstrate visual attention [Gre+15; MHG+14]. We followed the procedure suggested by [MHG+14] for generating both datasets, using the publicly available code[1]. The generated images are 80×80, covered by clutter, and the main content is located at a random location within the image boundaries. Both datasets include 500K training and 100K test images. Visual samples from these datasets are shown in Figure 5.6.

Implementation Details

Our action space consists of 10 transformations, including $\pm 4\ pixels$ translation in *x* and *y* direction, scaling of 0.8 in *x*, *y*, and *xy* direction (since the transformation is applied in a backward mapping manner, $scale < 1$ has a zoom-in effect), rotation of ± 10 degrees

[1] https://github.com/deepmind/mnist-cluttered

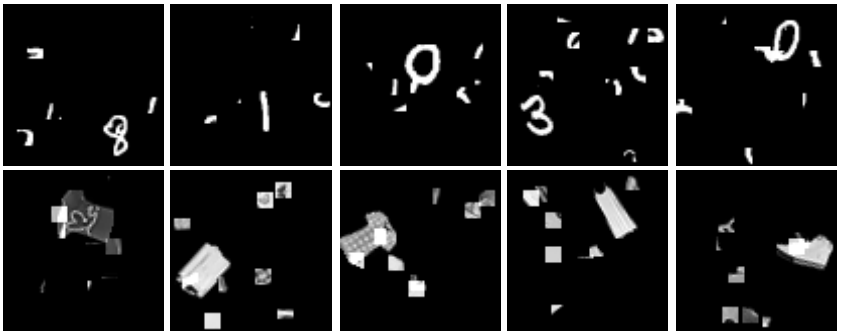


Fig. 5.6: Data samples from the cluttered MNIST in the top row and cluttered Fahison-MNIST in the second row.

Method	CMNIST(%)	CFMNIST(%)
MLP classifier	54.01	30.74
MLP classifier with STN	94.49	62.54
LeNet policy using PG algorithm	91.04	58.70
LeNet+LSTM policy using PG algorithm	95.88	70.27
LeNet+LSTM policy using AC algorithm	**96.83**	**71.61**

Tab. 5.1: Experiments with MLP classifier and STN as the baseline, followed by the SSTN approach results. For all the experiments, the classifier architecture is the same. The episode length for SSTN is set to 40. Our experiments cover the PG algorithm with different architectures as well as applying the AC algorithm on the best architecture. CMNIST and CFMNIST columns are classification accuracy for cluttered MNIST and Fashion-MNIST datasets, respectively. We observe that with the proper definition of the state space, our approach outperforms the STN method.

and *Identity* transformation. In Section 5.3.2, we show a comparison of the classification accuracies using different reward definitions, and Tables 5.1 and 5.2 present results using reward definition in Equation 5.17.

We evaluate our model in the following settings. First, we take a 2-layer fully connected network as the classifier (referred to as MLP in the following). The reason for choosing this simple classifier is to examine the improvement in classification accuracy based on only image transformations. Keeping the classifier architecture unchanged, we experiment with different policy architectures, including LeNet and LeNet combined with LSTM. We also experiment with both PG and AC training algorithms.

For comparison, we implement our own version of the STN model and train it on our dataset (the size of cluttered MNIST images is 60×60 in STN paper). For the localization network in Figure 5.4, we use the same LeNet architecture as in the policy network. In STN paper [JSZ+15], they used SGD as optimizer; however, we reached better results using the Adam optimizer, and we report the best-observed performance.

Method	CMNIST(%)	CFMNIST(%)
LeNet classifier	95.94	72.40
LeNet classifier with STN	97.72	77.38
LeNet policy using PG algorithm	95.82	74.59
LeNet+LSTM policy using PG algorithm	98.23	83.16
LeNet+LSTM policy using AC algorithm	**98.29**	**83.27**

Tab. 5.2: Experiment results when using LeNet for the classifier network and the episode length of 40. CMNIST and CFMNIST columns are the accuracy results for cluttered MNIST and Fashion-MNIST datasets, respectively.

The utilized LeNet architecture consists of two convolution layers with 32 and 64 kernels, followed by max pooling and two fully-connected layers with ReLU non-linearity. In the policy network, the last layer is a softmax, which generates the action probabilities. The actions are sampled from a *Categorical* distribution fitted to the softmax output. For all experiments, we use Adam optimizer with a learning rate of 10^{-4}, and the episode length is 40. In the next experiment setup, we change the classifier to LeNet and repeat similar experiments.

Experimental Results

Tables 5.1 and 5.2 show the results of our experiments using MLP and LeNet classifiers. We note that the impact of both the policy network in SSTN and the localization network in STN is only to transform the image before feeding it to the classifier and not to increase the power of the classifier.

For the policy network, first, we use a LeNet architecture and then combine it with LSTM. We aim to investigate if considering the state as the current single image satisfies the Markovian assumption. Based on the experiments with the LSTM module, we observe that this is an essential element, and the single image does not include all the required information. The reason is that the RL agent is supposed to learn the sequence of actions constructing the optimal affine transformation; therefore, it needs the tool for remembering the order of applying actions. The input to LSTM is the extracted feature map from the current transformed image, concatenated with one-hot encoded previous action. Finally, we take the best architecture from these experiments and train it with the AC algorithm. In the AC algorithm, we use the same network as the policy for the critic. Although it is possible to share weights between the actor and the critic, it is more stable if separate networks are used [Bah+16]. As expected, applying the AC training algorithm leads to further improvement; since it addresses some of the shortcomings in PG, as mentioned in Section 5.2.1. As results in Table 5.2 show, the LeNet classifier serves as a strong baseline

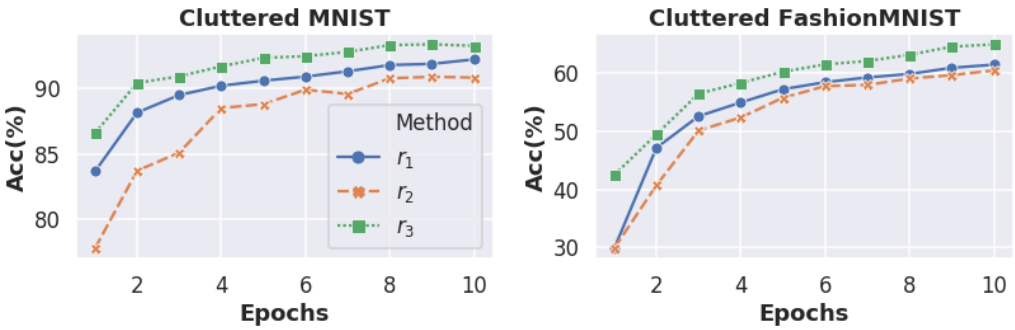


Fig. 5.7: Comparison between three different reward definitions using MLP classifier and LeNet+LSTM policy network using PG algorithm and episode length of 20.

and achieves high accuracy, especially in cluttered MNIST dataset. However, we still can get an improvement by modifying the input image before classifying.

Ablation Studies

In this section, we present an ablation study on the impact of the reward design and episode length on performance.

Reward: Figure 5.7 illustrates the epoch-accuracy curve depicting the performance of different reward definitions. While the performance is relatively similar across the different definitions, it is noteworthy that r_3 exhibits better performance compared to the others. We believe this behavior is because r_3 provides more concise information about the action taken compared to the discrete reward in Equation 5.15. Furthermore, we observe that the performance using r_2 is comparatively poorer than the other reward definitions. We posit that in rewards r_1 and r_3, we account for the change resulting from taking an action between every two time steps. In contrast, r_2 solely considers the loss value at the current time step without considering the change. These results suggest that r_2 incorporates less comprehensive information in comparison to the other two formulations.

Episode Length: Another important hyper-parameter is the number of time steps per episode. Figure 5.8 shows the performance of AC and PG algorithms for different time steps. As the results illustrate, the accuracy is better for more extended episodes. However, this can be seen as a trade-off between speed and accuracy. Another observation is that when using LeNet as the classifier, the performance of PG and AC algorithms are very similar. This indicates that using a stronger classifier decreases the variance in the reward signal.

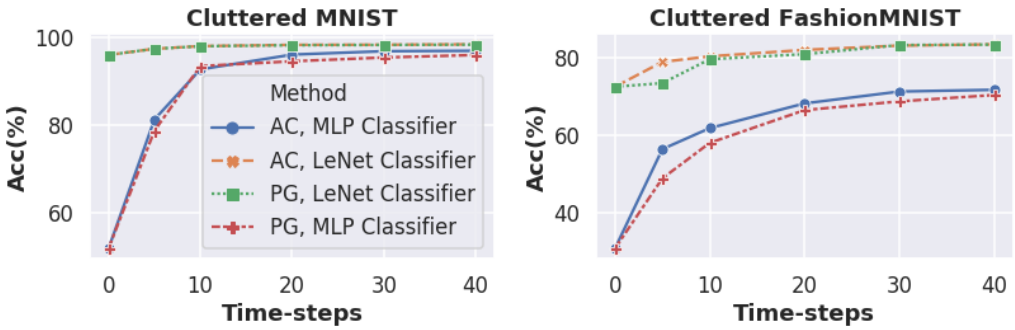


Fig. 5.8: Results for AC and PG algorithms with MLP and LeNet classifiers using different episode lengths.

5.3.3 Summary

In this work, we study cluttered image classification, a scenario where images contain a substantial amount of object-like clutter. To recover the classifier's performance in this challenging scenario, we develop a solution based on training an agent that learns to focus on the main content in the image via applying a series of geometric transformations such as translation and zooming. The training process of the agent relies on reinforcement learning, utilizing various reward functions that aim to maximize classification accuracy or minimize classification loss. To effectively capture the Markovian property crucial for the reinforcement learning pipeline, we propose an LSTM-based architecture that enables the model to summarize the selected actions in the previous time steps. We show the effectiveness of our model with extensive experiments on cluttered MNIST and Fashion-MNIST datasets, considerably outperforming the baselines.

5.4 Spatial Transformer Networks for Curriculum Learning

The learning process of intelligent beings such as humans and animals depends on many factors, one of which is the method of teaching and the way the information is presented to the learner. Numerous findings in the area of cognitive science demonstrate the importance of teaching designs and shaping the way the tasks are presented to the student [Ski58; Pet04]. Inspired by these findings on the learning mechanisms of humans, researchers in the machine learning community have explored the potential of utilizing similar ideas to improve the training process of neural networks [Elm93; Ben+09; San94; SSB85] to achieve faster convergence, find better local minima, and improve generalization. This line of research is referred to as curriculum learning.

There is a wide range of approaches addressing various aspects of this field in different applications. Rohde et al. investigated the effect of starting from easy examples in the task of language acquisition [RP99] and in [Ben+09], *Bengio et al.* studied where and when curriculum learning can be beneficial for the training process in the context of shape recognition tasks. Multiple efforts have looked into the impact of hard mining [CLM17; Zha+17; SGG16; SU07], which is finding the more challenging samples from the network's perspective and assigning them a higher weight. In [WCA18; HW19], the authors proposed various ways to sort the data samples based on an increase in difficulty. The goal is to measure the hardness of each task and develop a schedule or sampling method that initially exposes easier samples to the network.

Unlike the previous methods that focus on developing methods for distinguishing between the easy and hard samples [HW19; WCA18], we intend to directly manipulate the data distribution and generate the simpler tasks required for curriculum learning. Motivated by the success of Spatial Transformer Networks (STNs) [JSZ+15; Azi+19] in image classification, we take a new approach toward the preparation of easy and hard data samples for curriculum learning. STNs learn affine transformations that modify input data such that it is *easier* for the network to classify. In this work, we develop an approach to utilize the learned transformations in favor of curriculum learning. We highlight that by easier data distribution, we mean one that obtains a higher classification accuracy when using an identical classifier architecture and network capacity. We hypothesize that the transformation by STN simplifies all the intricacies of the original distribution that are caused by noise and clutter (see Figure 5.9). The motivation is that the network first learns on data with a high signal-to-noise ratio in the hope that, when the hard, noisy samples come, the model will be inclined to ignore artifacts caused by the noise itself. It is worth remarking that we utilize the data

Fig. 5.9: Data samples from cluttered MNIST dataset in the top row and the data transformed by SSTN in the second row.

transformed by STN during the training but do not use the STN during the test phase, which allows us to have a simpler inference framework.

We specifically use Sequential STNs (SSTNs) [Azi+19], as they allow us to access a range of data that varies in difficulty, from the original image to all incremental transformations thereof (Figure 5.5 illustrates the mechanism of the SSTN). Having a series of slightly modified and improved images at each time step, we hypothesize that this provides a hard-to-easy data distribution spectrum. We aim to use this characteristic in a curriculum learning setup to improve the training process of the classifier. We introduce two main approaches that exploit SSTN-processed data. The first variant, called Mixed-batch training, includes a portion of the SSTN-processed data with each mini-batch during training. The rationale is that the network can learn an embedding that is robust to the clutter of the original data by concurrently optimizing on the decluttered samples produced by the SSTN. In the second approach, called Incremental Difficulty training, optimization begins by exclusively using data that has been fully processed by the SSTN (i.e., data has undergone the maximum number of transformation steps). Then, we gradually increase the difficulty of samples being used for training by decreasing the number of transformation steps that the SSTN applies to each image (zero steps correspond to the original data).

5.4.1 Mixed-batch Curriculum Learning

As discussed earlier, transforming the input data through SSTN shifts the data distribution of the input such that it is easier to classify (higher classification accuracy) compared to the original data. Therefore, we measure the extent to which this easier distribution can be used as part of a curriculum for image classifiers.

In the mixed-batch strategy, we investigate whether mixing the original input images with a portion of data transformed by SSTN can facilitate the training process of the classifier. The

hope is that by presenting the input samples from the SSTN-processed and original data, the classifier can learn a representation that ignores the existing clutter and picks on the main content of the image. In this regard, we explore two different mixing strategies as described in the following.

In the first variant, we allocate a fixed portion of each mini-batch of input images to SSTN-processed data throughout the training. Formally, assume $X = \{x_i\}_{i=1}^{N}$ refers to the original data distribution with N data samples. Consider f_t as the transformation that is selected by the SSTN at the tth time step, and x_i^T is the product of sequentially applying T discrete transformations to x_i. We refer to the modified data distribution as $\hat{X} = \{\hat{x_i}\}_{i=1}^{N}$ where $\hat{x_i} = f_T \circ f_{T-1} \ldots \circ f_1(x_i)$ and T is the maximum number of transformation steps. In Mixed-batch setup, each mini-batch B_j consists of a fixed portion from the original and the modified data distribution: $B_j \subset X \cup \hat{X}$.

Moreover, we experiment with the Mixed-batch idea in a slightly different fashion. Instead of using a fixed share of SSTN-processed data in each mini-batch, we start the training with each mini-batch fully transformed by SSTN; then, we gradually increase the ratio of original-to-easy samples as training progresses such that towards the end of the training, only the original data are used. The rationale is to start with a high portion of easy samples and incrementally move toward the original data distribution, which will be used during the inference. We refer to this scheme as Dynamic Mixed-batch training.

5.4.2 Incremental Difficulty Curriculum Learning

We note that the SSTN performs a series of small transformations toward the final processed results. This allows us to have access to a spectrum of data distributions ranging from fully processed images (an easier distribution with higher classification accuracy) to original unprocessed data.

From the curriculum learning literature, we know that it is beneficial to the training of the neural networks to start from simpler tasks and progressively increase the complexity. In the Incremental difficulty setup, we identify a potential application of SSTN for generating this series of tasks or data distributions. Having T step transformations in SSTN, we start training the classifier with fully processed images and gradually decrease the number of transformation steps as the training progresses toward the original data. Formally, we sample each mini-batch B_j from the dataset $X' = \{f_T \circ f_{T-1} \ldots \circ f_1(x_i)\}_{i=1}^{N}$ where the number of transformation steps T gradually decreases from an initial maximum value to 0, resulting in the original dataset X. In this regard, we experiment with various scheduling strategies, including linear, cosine annealing, and exponential decay, where each method changes the data distribution at a different rate.

Method	Cluttered MNIST(%)	Cluttered Fashion-MNIST(%)
Baseline	87.9	72.6
Baseline*	91.2	83.6
Mixed-batch	89.9	83.5
Dynamic Mixed-batch	94.7	83.7
Incremental Difficulty	**95.0**	**84.9**

Tab. 5.3: Comparison of our proposed curriculum learning strategies with the baselines. Baseline and Baseline* rows present the results of training the classifier network without and with data augmentation, respectively. The results presented in this table are with the best-found hyperparameters regarding the portion of SSTN-processed data in the Mixed-batch approach and the number of transformation steps in Incremental Difficulty training.

5.4.3 Experimental Setup

In the following, we discuss the implementation details and provide the experimental results on Cluttered MNIST and Cluttered Fashion-MNIST datasets described in Section 5.3.2. Additionally, we include ablation studies to analyze various aspects of the proposed approach.

Implementation Details

The policy network consists of three convolution layers followed by an LSTM [HS97] module, a fully-connected layer, and softmax activation, which generates the probability distribution of the optimal action selection. The number of kernels in the convolution layers is 32, 64, and 64, respectively. The action-set employed in SSTN consists of 8 distinct transformations, including a translation of ± 4 pixels in vertical and horizontal directions, scaling up with a factor of 1.2, clockwise and counter-clockwise rotation of 10 degrees, and Identity transformation. Each sequence of transformations consists of 40 transformation steps ($T = 40$) unless mentioned otherwise, as we obtained the best results with this configuration. We use the Reinforce algorithm to train the policy. For more details on training the policy network, please refer to [Azi+19].

For the experiments on cluttered MNIST, the classifier architecture consists of a single convolution layer with 64 kernels followed by two fully-connected layers, referred to as LeNet1. The classifier used for the experiments on cluttered Fashion-MNIST consists of two convolution layers with 32 and 64 kernels and two fully-connected layers, referred to as LeNet2. For all the experiments, we use the Adam optimizer [KB14], a learning rate of 10^{-4}, and a batch size of 64.

Experimental Results

Table 5.3 presents the experimental results on cluttered MNIST and Fashion-MNIST datasets. For the baseline, we trained the model with and without data augmentation with the standard training process (Adam optimizer and a learning rate of 10^{-4}, trained until convergence).

For the Mixed-batch experiment, we tried different setups, including varying portions of SSTN-processed data in each mini-batch. The best-found result is recorded in Table 5.3, and the ablation on this hyperparameter is presented in Table 5.6. In Dynamic Mixed-batch, initially, each mini-batch is composed of the images transformed by SSTN (easier). Then, we gradually reduce the portion of easy-to-original by a rate of one sample per epoch. In the Incremental difficulty experiment, we start the training with the data processed by SSTN for a maximum of T time steps. The number of transformation steps is decreased by one every five epochs, slowly shifting the data distribution from easy towards the original dataset.

Although both cluttered MNIST and Fashion-MNIST inherently include affine transformations (as it is used in the generation of these datasets), we observed that applying further affine transformation during the training in the context of data augmentation is effective, especially in the case of cluttered Fashion-MNIST. According to the results in Table 5.3, we see that the Mixed-batch variant performs merely as well as standard data augmentation; however, Dynamic Mixed-batch achieves substantially better performance compared to the Mixed-Batch. Furthermore, we observe that the Incremental Difficulty training considerably improves the performance compared to the baselines. From these results, we conclude that it is important to finalize the training of the neural networks with the data distribution which is to be used at the inference phase. We note that the accuracy improvement is more pronounced in the case of cluttered MNIST dataset; we believe this is because the underlying SSTN trained on cluttered MNIST does a better job of processing the input and removing the clutter from the image. This is manifested in the higher performance gain achieved when using SSTN for cluttered MNIST compared to the Fashion-MNIST.

In Table 5.4, we present the results for Incremental Difficulty training when experimenting with three different scheduling methods. These functions are used to determine the rate at which the number of transformation time steps is dropped from T to 0. In Linear scheduling, the number of transformation steps is reduced by one every five epochs. The transformation steps in Cosine Annealing follow $\lfloor \cos(\frac{epoch.\pi}{2T}) \rfloor$ and Exponential Decay schedulers follows $\lfloor \exp(-\frac{epoch.T}{\tau}) \rfloor$, where the hyperparameter τ determines the decay pace and is set to 30. We obtained the best results using simple Linear scheduling.

Method	Cluttered MNIST(%)	Cluttered Fashion-MNIST(%)
Linear Decay	**94.8**	**84.9**
Cosine Annealing	93.7	83.1
Exponential Decay	94.7	84.7

Tab. 5.4: The impact of three different scheduling functions on the Incremental Difficulty training. The number of transformation steps T is set to 20 in this set of experiments.

Ablation Studies

In this part, we provide an ablation on the role of the number of transformation steps T in the Incremental Difficulty as well as the impact of the portion of SSTN-processed data in Mixed-batch curriculum learning. In Table 5.5, we present the results for the Incremental Difficulty method with various T values (maximum number of transformation steps). We see that the model performance converges with about 20 transformation steps.

Time-steps	Cluttered MNIST(%)	Cluttered Fashion-MNIST(%)
T : 1	92.6	83.1
T : 10	94.0	84.4
T : 20	94.8	**84.9**
T : 40	**95.0**	**84.9**

Tab. 5.5: An ablation on impact for T, the maximum number of transformation steps in our Incremental Difficulty approach.

In Table 5.6, we experiment with different ratios of easy-to-original data in the Mixed-batch training approach. In this approach, we obtained the best results by including 4 and 16 SSTN-processed samples in a mini-batch of 64 images in cluttered MNIST and Fashion-MNIST datasets, respectively.

Ratio	Cluttered MNIST(%)	Cluttered Fashion-MNIST(%)
4/64	**89.9**	82.9
8/84	89.3	83.2
16/64	88.6	**83.5**
32/64	86.0	82.9

Tab. 5.6: Ablation on the Mixed-batch experiment, when using a different ratio of SSTN-processed data in a mini-batch of size 64.

5.4.4 Summary

In this section, we present a new approach for curriculum learning by employing SSTN [Azi+19] to generate the easy-to-hard task ordering, which is known to be advantageous to the training of neural networks. STNs have been successful in de-cluttering the input

image and modifying the input image such that it is easier for the network to classify, i.e., the classifier achieves higher accuracy [JSZ+15; Azi+19]. In this work, we use this characteristic to design a curriculum learning strategy without the need to manually define an a priori ordering of training data by allowing the SSTN to modify all or part of the training data according to a predefined strategy. We apply these methods to the problem of training a classifier for cluttered versions of the MNIST and Fashion-MNIST datasets and show that they increase the classifier accuracy for a given architecture. We observe that the gained improvement was more significant in the case of the cluttered MNIST, potentially due to having a more effective SSTN policy for this dataset. Therefore, a future research direction that can directly benefit our work would be improvements in the design of the SSTN algorithm.

5.5 Image Classification in the Wild

In this section, we study image classification in the wild: a realistic scenario where there are multiple object instances in the scene, and we are interested in classifying the image based on the most salient object. Image classification has been a standard task in the vision community for developing and improving deep learning architectures; however, the majority of architectural advances [KSH12; He+16; Xie+17; Sze+17; Dos+20] evaluate the model performance on highly curated images such as CIFAR [KH+09], or Imagenet [Den+09]. As such, a side-product of focusing on increasing the model accuracy on specialized and curated datasets is the lack of out-of-domain generalization [HD19b] and poor performance on images where the main content is not located at the image center, let alone if the scene is cluttered [JSZ+15].

In this work, we investigate *salient object classification* on challenging PASCAL VOC dataset [Eve+10], inspired by Spatial Transformer Networks [JSZ+15; Azi+19]. Our main hypothesis is that zooming in on the salient object and cropping out secondary objects regarded as clutter is beneficial for the classifier. Hence, increasing the Intersection over Union (IoU) of the salient object (i.e., the ratio between the salient object's bounding box area and the image area) can be used as a training signal. Thanks to the availability of object detection information in PASCAL VOC [Eve+10], we use bounding-box information to compute the IoU of different object instances in the scene. Instead of relying on human annotations for identifying the salient object, we resort to the approximate assumption that the object size serves as an indicator of object saliency and consider the largest object as the salient one.

Specifically, we build on top of the proposed SSTN algorithm [Azi+19], as the RL-based solution in [Azi+19] offers a flexible framework that allows for employing various differentiable and non-differentiable training objectives. We advance the algorithm proposed in [Azi+19] by employing Q-Learning, an effective algorithm for scaling to high dimensional input data such as images [Mni+13]. Moreover, we propose reward-shaping functions that attempt to directly increase the IoU of the salient object, resulting in improved classification accuracy. Our method, named DQ-SSTN, increases the IoU of the salient object by 11.31 pp and the overall classification accuracy by 1.82 pp. We observe that our method is especially effective for smaller objects (objects that cover less than 20 percent of the image area), where we obtain an improvement of 3.63 pp in accuracy.

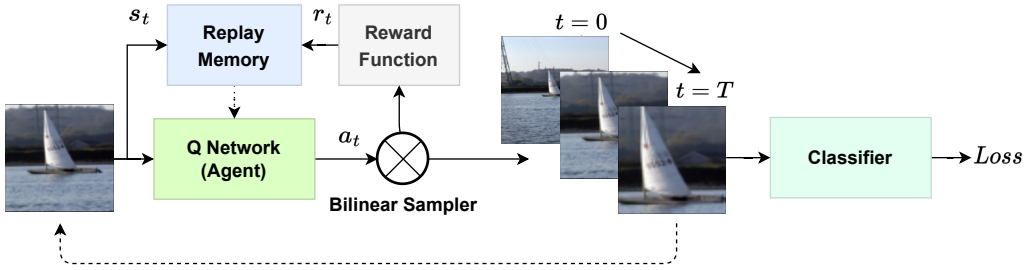


Fig. 5.10: The overall architecture of DQ-SSTN. Our model sequentially modifies the input image by applying a series of simple and discrete transformations (a_t) selected by an agent trained to maximize the overall obtained reward (r_t).

5.5.1 Sequential Spatial Transformer Networks for Salient Object Classification

In this section, we elaborate on our proposed approach toward salient object classification, where the object saliency is determined based on the object's bounding box area. Our goal is to extend the SSTN algorithm [Azi+19] for working with real-world images with multiple object instances present in the scene. While SSTN uses Policy Gradient for training the RL agent, we propose employing the Q-learning algorithm, as recent works [Mni+13] confirm the effectiveness of this algorithm for better scaling to higher-dimensional data. We experiment with various reward functions that, based on different metrics, aim to boost the performance of the downstream classifier. Thanks to the RL-based framework, our model is flexible to work with non-differentiable training objectives. Figure 5.10 visualizes the overall architecture of our model.

As explained in Section 5.2.1, the first step in formulating an RL-based problem is to define state space, action space, and reward functions required to describe the MDP framework. For the **state space** s_t, we consider the transformed image at time t, which has undergone a sequence of transformations (the actions selected by the agent). Our **action space** a_t consists of 6 discrete affine transformations with fixed parameters, including translation in 4 directions, zooming, and *identity*. The identity action allows for batch training; as such, we use a fixed number of transformations T, and the agent can decide if specific images in the batch are already sufficiently transformed by choosing the identity action.

Regarding the **reward** r_t, we initially experiment with the same functions proposed in [Azi+19] referred to as *Continuous loss-reward* and *Discrete Acc-reward*. *Continuous loss-reward* is the difference between the classification loss before and after applying the selected action (hence the reward is positive when reducing the loss value, Equation 5.17). *Discrete Acc-reward* rewards the agent with $+1$ if the classifier's prediction changes from incorrect to correct or -1 vice-versa (Equation 5.15).

Algorithm 2: DDQN training algorithm for the proposed DQ-SSTN architecture illustrated in Figure 5.10 (one epoch).

Input : Images and classification labels
Output : Classifier and sequential spatial transformer policy

Initialize classifier (ψ)*, Q network* (θ)*, and target network* (ϕ)*,*
$Replay\_Memory = [empty]$
for *Image = 1 : N* **do**
  $s = s_0,\ t = 0$
  while *t < episode-length T* **do**
    # Select action through ε-greedy policy
    if $p < \varepsilon + (1 - \varepsilon) \cdot e^{-\frac{n}{d}}$ **then**
      random a_t
    else
      $a_t =$ argmax $Q^*(s, a)$
    end
    $s_{t+1} = \boldsymbol{apply\_transformation}(s_t, a_t)$
    $r_t = \boldsymbol{compute\_reward}(s_t, s_{t+1})$
    $Replay\_Memory.\boldsymbol{append}(s_t, a_t, r_t, s_{t+1})$
  end
  $pred = classifier.\boldsymbol{predict}(s_T)$
  $\psi = \psi - \alpha \cdot \nabla_\psi \boldsymbol{CrossEntropy\_loss}(pred, label)$ # update *classifier*
  $s_i, a_i, r_i, s_{i+1} = Replay\_Memory.\boldsymbol{sample}()$ # update *Q network*
  Q_θ = *Q network*.$\boldsymbol{action\_values}(s_i, a_i)$
  $\Gamma = r_i + \gamma \cdot \max_{a'}(\text{DQ-SSTN}.\boldsymbol{rate}(s_{i+1}, a'))$ # Bellman optimality equation
  $\theta = \theta - \alpha \nabla_\theta \boldsymbol{HuberLoss}(Q_\theta,\ \Gamma)$
  if $iteration \mod n \equiv 0$ **then**
    $\phi \leftarrow \theta$
  end
end

We further explore the possibility of improving the classifier accuracy by learning to increase the IoU of the salient object. To this end, we positively reward the agent when the selected action results in increasing the area of the salient object. This way, the agent is encouraged to zoom around the salient object. This reward named *Continuous IoU-reward* is defined as:

$$r_t^{(c\_iou)} = \text{IoU}_t - \text{IoU}_{t-1} \tag{5.18}$$

Moreover, we experiment with the discrete version of the IoU and refer to it as *Discrete IoU-reward*:

$$r_t^{(d\_iou)} = \begin{cases} \mathbf{+1} & \text{if IoU}_t > \text{IoU}_{t-1}, \\ \mathbf{-1} & \text{if IoU}_t < \text{IoU}_{t-1}, \\ \mathbf{0} & \text{else.} \end{cases} \tag{5.19}$$

Finally, we experiment with a weighted combination of the IoU-based and loss-based reward functions defined as:

$$r_t^{(combined)} = \alpha \cdot r_t^{(c\_loss)} + \beta \cdot r_t^{(d\_iou)} \tag{5.20}$$

where α and β are hyperparameters. Note that the IoU-based and discrete loss-based rewards are non-differentiable objectives that can be optimized within the RL-based problem formulation.

We train our model using the DDQN algorithm [Mni+13], explained in Section 5.2.1. As proposed in the DDQN algorithm, we deploy a replay memory for storing the state-action transitions (see Figure 5.10). Subsequently, uncorrelated training samples are randomly drawn from this memory. Further, a target network [Mni+13] responsible for predicting the expected reward is employed to break the inter-dependency in the Bellman update rule (see Equation 5.10). During the training phase, the actions are selected following the ε-greedy policy described in Equation 5.11. The overall training algorithm is presented in algorithm 2.

5.5.2 Experimental Setup

In this section, we explain the characteristics of the dataset used for training and validation, followed by a discussion on implementation details. Next, we present the evaluation results, followed by several ablation studies aiming at better understanding the model behavior.

Dataset

PASCAL VOC [Eve+10] is a real-world dataset with pictures containing multiple objects from 20 object classes. The dataset provides the bounding box and the category of each object. We combine the dataset versions 2007 and 2012 to get as many images as possible, resulting in a training set of 8218 and a test set of 8333 frames.

For the classification ground-truth label, we consider the category of the object with the largest area. Consequently, the *Top-1* accuracy depends on whether the prediction corresponds to the largest visible object. As additional metrics, we use the *Top-2* and *Any* accuracies for evaluation (we only optimize over Top-1 accuracy). The *Top-2* accuracy additionally allows the prediction to be the second largest object, and the *Any* accuracy permits the prediction to match any object present within an image. We highlight that using the PASCAL VOC dataset for single-class labeling results in a non-uniform class distribution, where class *person* has more than 1700 images compared to other classes,

which range from below 200 to 600 images per class. We discuss a possible bias towards the dominant class in Section 5.5.2.

Implementation Details

The backbone of the classifier and the Q-network consists of a ResNet18 [He+16] where the last fully-connected layer is modified to match the number of the object classes and the number of actions, respectively. ResNet18 is an architecture quite successful in image classification tasks, and the library PyTorch [Pas+17] provides pre-trained weights. We downscale the input images to 224×224 to match the ResNet18s implementation and perform horizontal flipping augmentations to increase the variety in the dataset.

In the Q-Learning objective (see Equation 5.10), we use $\gamma = 0.95$ and update our target network after 100 agent updates. Our replay memory stores 1000 transitions. The ε-greedy strategy Equation 5.11 uses $d = 50000$ and $\varepsilon_{\text{end}} = 0.05$. The DQ-SSTN action space includes a translation of 4 pixels in each cardinal direction and zooming-in by a factor of 0.8, with a fixed number of transformation steps $T = 10$. In the corresponding experiments, the weight factors in Equation 5.20 are set to $\alpha = 1$ and $\beta = 0.8$. We train our model with Adam [KB14] optimizer and a learning rate of $5 \cdot 10^{-6}$ for 50 epochs.

Experimental Results

Table 5.7 provides a comparison between the baseline classifier without DQ-SSTN and our proposed DQ-SSTN method using different reward functions. As can be seen from the results, the best accuracy was obtained employing the discrete IoU reward, improving *Top-1* accuracy by 1.82 pp, *Top-2* accuracy by 1.86 pp and *Any* accuracy by 1.23 pp, respectively.

Method	**Top-1 (%)**	**Top-2 (%)**	**Any (%)**
Baseline classifier	80.04	82.97	88.22
Continuous loss-reward (r^{c_loss})	80.84	83.75	88.83
Discrete Acc-reward (r^{d_acc})	80.43	83.61	88.67
Continuous IoU-reward (r^{c_iou})	80.15	83.09	88.19
Discrete IoU-reward (r^{d_iou})	**81.86**	**84.83**	**89.45**
Weighted combination-reward ($r^{combined}$)	81.54	84.56	89.27

Tab. 5.7: Comparison of classification accuracy of the baseline with our method when using different reward functions, on PASCAL VOC [Eve+10]. We obtained the best results with the discrete IoU-based reward signal.

In Figure 5.11, we provide an analysis of the IoU and classification accuracy before and after applying the transformations selected by our DQ-SSTN model. To better understand the

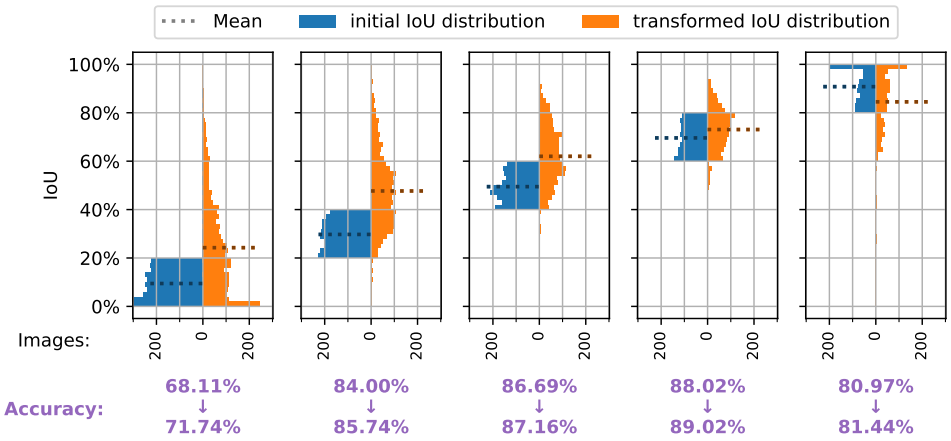


Fig. 5.11: IoU and classification accuracy before and after applying the DQ-SSTN transformations. The dataset is split into five bins according to the initial IoU. Classification accuracy changes are noted under each column. DQ-SSTN is especially useful for smaller objects (first bin) where the classification accuracy is improved by 3.63 pp.

impact of the transformations on objects with different sizes, we divide the test-set images into 5 bins, considering the initial IoU. Applying the DQ-SSTN transformations leads to an average increase of 11.31 pp in the target class IoU (the dotted line in this figure shows the average IoU per bin). Furthermore, we evaluate the classification accuracy for each bin before and after applying transformations. We observe that our model is more effective in improving the classification accuracy of smaller objects, increasing it by 3.63 pp for objects with an IoU less than 20%. Interestingly, the IoU of the right-most bin decreases while the accuracy increases; this behavior will be observed in Figure 5.13 again and discussed in Section 5.5.2.

Visual examples of the transformations learned by our model are illustrated in Figure 5.12.

Hard Mining. Our derived version of the PASCAL VOC dataset adapted for classification suffers from a considerable imbalance in both class distribution and IoU distribution (see Section 5.5.2). This imbalance may hurt performance as the learned solution could be biased towards the dominant category. Hard mining is a training technique that has been proven effective [SGG16; DGZ17] to alleviate the effect of data imbalance by assigning a higher weight to underrepresented (therefore more challenging) data samples.

To this end, we experiment with multiple hard-mining strategies and provide the results in Table 5.8. As our overall objective consists of the classification and the reward maximization terms, we can perform hard mining by re-weighting the classifier's loss or the reward of the more challenging data samples.

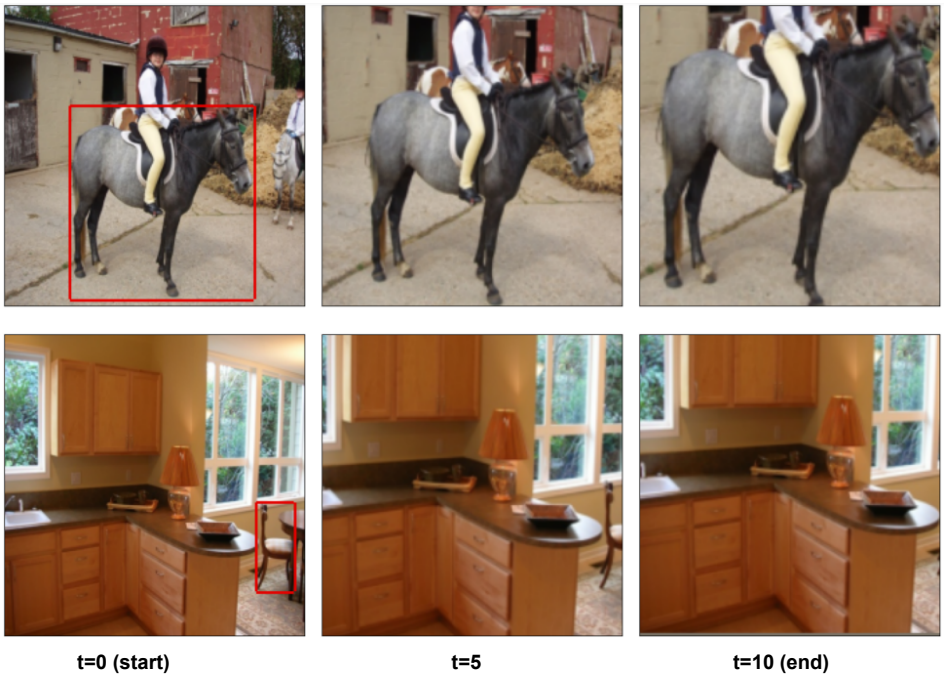


Fig. 5.12: Visual examples of our DQ-SSTN model gradually focusing on the salient object. The bounding box of the largest object is visualized in red in the starting frame (t=0).

Initially, we consider IoU as a measure of a data sample hardness. Therefore, we reweight either the loss function or the reward signal by the inverse of IoU, assigning higher importance to samples with smaller IoU (*Weight reward by inverse IoU* and *Weight loss by inverse IoU*). Surprisingly, our results did not improve with this technique. Next, we reweigh the data samples based on the performance of the baseline classifier. If an image is classified incorrectly, we assign a weight of 1 and, if it's predicted correctly, a constant weight < 1. This assigns higher importance to those images that are more difficult for our classifier (*Weight reward by accuracy* and *Weight loss by accuracy*). The results show minor improvements of 0.06 pp for either Top-1 or Top-2 accuracy; hence, we do not find these techniques helpful to our algorithm.

Method	**Top-1 (%)**	**Top-2 (%)**	**Any (%)**
Baseline classifier	80.04	82.97	88.22
DQ-SSTN without hard mining	81.80	85.25	**89.72**
Weight reward by inverse IoU	81.40	85.13	89.60
Weight loss by inverse IoU	81.08	84.72	89.32
Weight reward by accuracy	**81.86**	84.83	89.45
Weight loss by accuracy	81.75	**85.31**	89.69

Tab. 5.8: The impact of different hard-mining techniques tried on the classification accuracy. Whilst the *weight by accuracy* methods improved the Top-1 or Top-2 metric, respectively, by 0.06pp, we do not consider this as a noteworthy improvement.

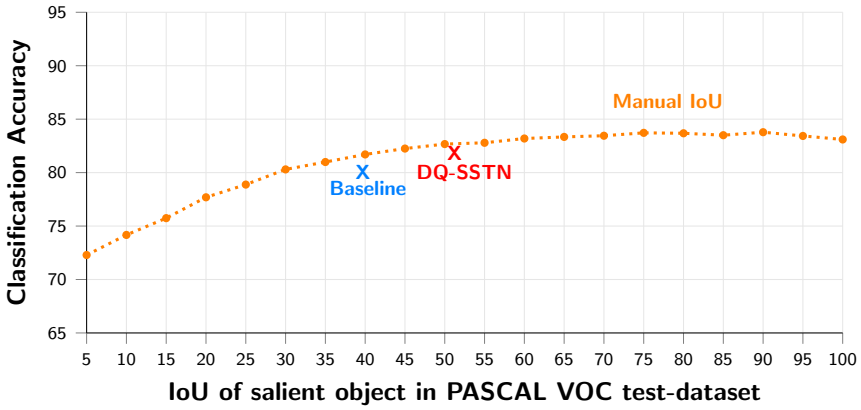


Fig. 5.13: Our experiment indicates a correlation between salient object IoU and classification accuracy (both in %). Zooming on the target object (increase in IoU) leads to better classification accuracy for this object.

Ablation Studies

In this section, we provide several experimental studies to: 1) Confirm the relation between the increase in an object's IoU and the improvement in classification accuracy, 2) Analyze the impact of dataset bias, and 3) Study the issues raised from the saliency assumption where we consider the largest object as the salient one.

Correlation of IoU and Accuracy. The underlying assumption in DQ-SSTN is that classification accuracy improves with increasing the target object's IoU. To confirm this, we run an experiment where we construct multiple datasets, each with a constant IoU enforced synthetically. To this end, we determine the salient object by the size of the bounding box and crop around it such that we achieve a fixed IoU for that object. Furthermore, we include random translations to hinder the network from cheating and getting biased based on the positioning of the object. This results in a dataset where each image's salient object has the same constant IoU.

After constructing 20 datasets with different IoUs, we train and evaluate a classifier on each dataset separately. As can be seen in Figure 5.13, an increase in the IoU indeed leads to improved classification accuracy. We observe that adjusting the IoU to more than 90% reduces the accuracy. This shows that extreme zooming has a negative impact on performance, as informative parts of the object can be displaced out of the classifier's focus.

Bias of the Person Class. One property of the PASCAL VOC dataset is an imbalance in the class distribution. Based on object size, about 20% of all images are assigned to *person* class. As a result, we observe that the attention of our DQ-SSTN is drawn towards persons: If a

(%)	pred. *person*	pred. other
person present	39.37	60.63
person not present	11.15	88.85

Tab. 5.9: The prediction of the classifier on images that were predicted incorrectly. If there is a person present, the DQ-SSTN has a high chance of focusing on it.

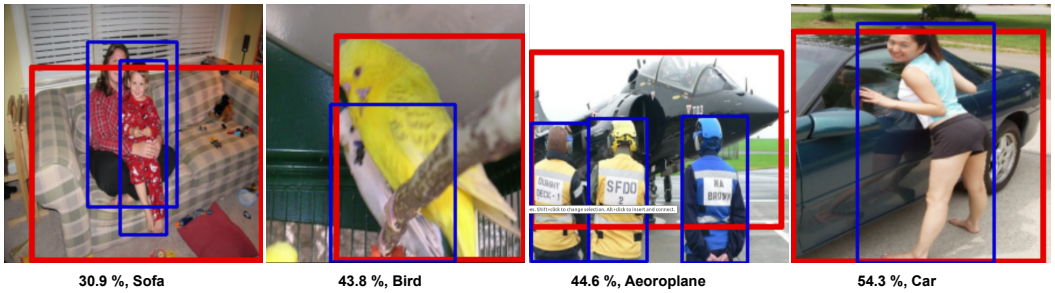


Fig. 5.14: Visual examples where the main object intersects with other objects. Red frames surround the largest object, and blue frames the secondary ones. The value below each image is the percentage of the largest (salient) object's bounding box that intersects with other bounding boxes, and the assigned true label is also given. The first image is an image below threshold $th = 0.4$ but still considered cluttered, and the second one is vice versa. The other two examples further highlight our choice of threshold.

person is present in the scene, the classifier is biased towards classifying the image as *person* category. In Table 5.9, we provide statistics for misclassifications considering whether a person is present as a secondary object. It is clearly visible that in cases where a person is present, the classifier is biased toward this class. However, if there is no person in the image, the classifier does not overly tend toward predicting one. Note that a random classifier would select *person* class around 20% of the time, following the dataset distribution in the training set. This is an inherent issue in the PASCAL VOC dataset, as there are far more person objects throughout the dataset than compared to other classes.

Issues with Saliency Assumption. In our setup, we assume the object with the largest bounding box to be the most salient one. Here, we investigate how often and to which extent this assumption is violated and how this impacts the performance of the DQ-SSTN.

An example in contrast with our saliency assumption is when a smaller object is located in front of a bigger one. Consider a person sitting on a sofa (as in Figure 5.14); in this scenario, humans consider the person as the salient object, while based on our assumption, the sofa is labeled as the salient category. Interestingly, we observed that in most cases, the classifier also predicts the front object as the correct class (*person* in this example), but this is considered a misclassification based on our evaluation.

To better understand this issue, we visualize the overlap characteristics of our data in Figure 5.15. In this figure, we see the portion of images in which the main object (largest/salient)

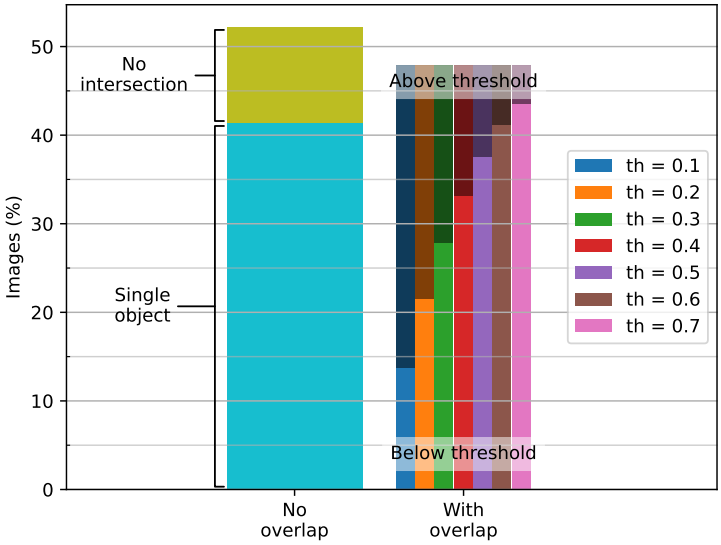


Fig. 5.15: This histogram shows the composition of our dataset regarding the relation between the most salient object and overlapping objects. Depending on the selected threshold, between $5\%(th = 0.7)$ to $35\%(th = 0.1)$ of all images are to some extent covered by another object instance, thus violating our saliency assumption.

is not overlapping with other objects in the scene, as well as the number of images in which the main object intersects with other object instances considering different overlap thresholds. For example, the column with $th = 0.5$ shows the number of images in which the overlap between the main and the secondary objects is lower or higher than 50% of the main object's area. We observed that 41.4% of the images only have one object, and 10.8% of all images are accompanied by secondary objects, but their bounding boxes do not intersect with each other. This means that in 52.1% of the images, our assumption about saliency strictly holds. However, in 47.89% of the images, the salient object is, to some extent intersecting with (and possibly being occluded by) another object. Considering the qualitative examples in Figure 5.14, we regard a threshold of $th = 0.4$ as critical; i.e., all images that have more than 40% of their bounding-box concealed by another object do not follow our saliency assumption based on object size. Based on this threshold, about 15% of the dataset conflict with our saliency assumption.

To quantitatively examine the impact of our saliency assumption on the DQ-SSTN performance, we divide the dataset into three categories; the images with the main object not covered by any other objects and the images in which the main object is covered by other instances more *or* less than 40% ($th = 0.4$ in Figure 5.15) We evaluate our model separately on each group and present the results in Table 5.10. As expected, we observe that

	Accuracy (%)
Total	81.86
No clutter	86.05
Cluttered, below $th = 0.4$	82.83
Cluttered, above $th = 0.4$	64.88

Tab. 5.10: DQ-SSTN classification accuracy on each subset of PASCAL VOC when categorized based on the overlap threshold of 0.4 (as shown in Figure 5.15).

the classification accuracy is significantly lower for the data in which the salient object has an overlap of over 40% with other objects in the scene.

5.5.3 Summary

In this work, we study the task of salient object classification. We introduce DQ-SSTN, a Sequential Spatial Transformer Network based on Deep Q-Learning. Our model learns to zoom on the salient object by iteratively selecting an affine transformation to increase the IoU of the largest object in an image. We experimentally demonstrate the effectiveness of our method in improving the classification accuracy, especially for smaller objects, where we achieve an improvement of 3.63 pp. Additionally, we conduct several ablation studies to thoroughly analyze the behavior of our model and identify potential failure scenarios. Through these studies, we uncover two primary limitations: the lack of specialized data and the consideration of size as saliency, along with the presence of dataset bias.

6 Video Object Segmentation

6.1 Introduction

Video Object Segmentation (VOS) is an active research area of the visual domain that aims at pixel-wise tracking of a set of target objects. One of its fundamental sub-tasks is one-shot VOS: given only the object segmentation mask in the first frame, the task is to provide pixel-accurate masks for the object over the rest of the sequence. VOS plays an essential role in various applications, such as video editing, autonomous driving, and robotics [Yao+20]. During the last years, a wide variety of learning-based solutions have been proposed for VOS trying to maximize the segmentation accuracy via addressing different challenging scenarios such as tracking smaller objects, handling occlusion, fast motion, and crowded scenes with similar object instances [Gao+22].

The current approaches in the literature can be roughly categorized into two main groups of propagation-based and matching-based methods. In propagation-based methodology, the object mask is sequentially propagated to the subsequent frames by adjusting the mask's location and appearance using visual and motion information. In contrast, matching-based techniques capture the target object's mask in each frame by finding the correspondences between the current frame and the provided object mask in the first frame. Each category has certain advantages and limitations. For example, propagation-based algorithms capture the object's location and appearance evolution over time and utilize the motion information; however, they suffer from error propagation as segmentation inaccuracy for each image impacts the subsequent frames. In correspondence matching, on the other hand, the object's mask at every image is captured independently by comparison to a set of reference templates (e.g., segmentation masks at $t = 0$). As a result, the model does not face the error propagation issue, but the performance degrades when similar objects appear in the scene or when the object's appearance considerably changes compared to the reference template.

In this chapter, we start with an in-depth study of one of the popular propagation-based baselines that uses RNNs for utilizing spatiotemporal information [Xu+18], as this model offers a straightforward design with good performance and fast inference. This model consists of an encoder-decoder that maps the RGB images to the segmentation mask using an RNN module in the bottleneck for tracking the target object. We identify two main limitations of this model for handling challenging video scenes for tracking smaller

objects and accurately segmenting the object boundaries. We empirically observe that this issue arises from spatial information loss in the bottleneck due to multiple downscale and pooling operations in the encoder network. Furthermore, segmenting the pixels around the object boundary is more difficult due to the potential mixup between the features from the object and the background. Accordingly, we improve this approach by proposing a model that manipulates multi-scale spatiotemporal information using memory-equipped skip connections. Furthermore, we incorporate an auxiliary task based on distance classification, which considerably enhances the quality of edges in segmentation masks, supported by qualitative and quantitative experimental results.

Although the proposed solutions mentioned above improve the segmentation quality for object boundaries and smaller objects, the model's performance significantly deteriorates when segmenting longer videos. This deterioration is mainly caused by the inherent limitations of the RNN module, namely limited memory and error propagation. We take inspiration from the matching-based VOS methods to address these issues and propose a hybrid architecture that employs a dual mask propagation strategy by fusing the spatiotemporal RNN features with correspondence matching. Our experiments show that augmenting the RNN with correspondence matching is a highly effective solution for adding extra modeling capacity to the model and reducing the drift problem. Extensive analysis of our model's behavior in challenging cases such as occlusion and long sequences shows that the proposed hybrid architecture significantly enhances the segmentation quality in these challenging scenarios.

In the last section, we study several aspects regarding the behavior of the proposed hybrid VOS model. First, we design a bidirectional architecture that additionally utilizes the information from the future frames to generate the object mask at each time step. Second, we investigate a multi-task training setup that combines the VOS architecture with unsupervised optical flow estimation. Since optical flow estimation and VOS are essentially similar tasks, we ask whether the VOS model can benefit from the additional flow information. Finally, we perform an extensive benchmark to quantify the impact of the design choice for each component in the hybrid VOS architecture. We evaluate the model performance with several convolutional and attention-based backbones, different variants of the RNN module as well as different designs for the fusion module responsible for combining the RNN and matching information.

The content in this chapter is based on the publications [Azi+21a], [Azi+21b], and [Azi+21c].

6.2 Related Work

A large body of research in Computer Vision literature has studied VOS during the last decade. The classical methods for solving VOS were mainly based on energy minimization [BM10; FI14; PF13; SSB15]. Brox *et al.* [BM10] propose a model based on motion clustering and segment the moving object via the analysis of the point trajectories throughout the video. They also use motion cues to distinguish foreground from background. Faktor *et al.* [FI14] present a method based on consensus voting. They extract the superpixels in each frame and, by computing the similarity of the superpixel descriptors, then use the nearest neighbor method to cluster the most similar superpixels together in a segmentation mask. [JG14] addresses the problem of fast motion and appearance change in the video by extending the idea of using superpixels to using super-voxels (adding the time dimension) and taking into account the long-range temporal connection during the object movement.

Since the advent of Deep Learning [KSH12], the Computer Vision community has witnessed significant progress in the accuracy of VOS methods [Man+18; Per+17; TAS17]. The success of learning-based methods can largely be accounted to progress made in learning algorithms [KSH12; He+16] and the availability of large-scale VOS datasets such as Youtube-VOS [Xu+18].

In one-shot VOS, there are two training schemes, namely offline and online training. *Offline training* is the standard training phase in learning-based techniques where the model's weights are learned during the training phase and kept fixed during the inference. As the segmentation mask of the first object appearance is available at test time, *online training* refers to further fine-tuning the model on this mask with extensive data augmentation. This additional step considerably improves the segmentation quality at the expense of slower inference. Earlier learning-based methods for solving VOS heavily relied on online training. In [Man+18], authors extended a VGG-based architecture designed for retinal image understanding [Man+16] for VOS. They start with the pre-trained weights on ImageNet [Den+09] and then further train on a specialized VOS dataset [Per+16a]. This model relies on online training and boundary snapping to achieve good performance. [VL17] further improves this method by employing online adaption to handle drastic changes in the object's appearance. Moreover, some earlier works rely on object detection algorithms such as RCNN [Ren+15; He+17]. For example, [LVL18] takes a multi-step approach, in which they first generate the region proposals and then refine and merge promising regions to produce the final mask. They additionally use optical flow to maintain temporal consistency. In [Li+17], the authors integrate a re-identification module for recapturing objects occluded or lost at some point in the video. These methods produce high-quality segmentation masks, but their design is complex, their inference time is relatively high, and they still require online training to perform well. However, more recent methods focus on the architecture

design and improved task modeling toward obtaining accurate segmentation masks without requiring the time-consuming online training phase [Wug+18; Joh+19]. This aspect is of high practical importance as more VOS applications require real-time performance. In the following, we review recent VOS approaches categorized into *propagation-based* and *matching-based* algorithms.

Propagation-based Video Object Segmentation. Methods relying on object mask propagation for solving VOS utilize motion information and visual cues from the RGB input image to propagate the mask from the current time step to the future frames, accounting for the shift in the object's location and appearance. To this end, [Xu+18; TAS17; Ven+19] employ various forms of RNNs for processing motion and computing the spatiotemporal features. Utilizing RNN enables the model to track the target object in a temporally consistent manner. Differently, Perazzi *et al.* propose MaskTrack, a VOS solution based on guided instance segmentation [Per+17]. They utilize a DeepLab architecture [Che+17a] and modify the network to accept the previous segmentation mask as an additional input. Therefore, a rough guidance signal is provided to the model to mark the approximate location where the object of interest lies. Yang *et al.* [Yan+18] take a meta-learning approach and train an additional modulator network that adjusts the middle layers of a generic segmentation network to capture the appearance of the target object. In [Zha+20b], the authors develop a model that propagates the segmentation mask based on an affinity in the embedding space. They propose to model the local dependencies by using motion and spatial priors and the global dependencies based on the visual appearance learned by a convolutional network.

To accurately propagate the object mask, MHP-VOS [Xu+19] generates multiple propagation Hypotheses based on the object tracks from the previous frames and the detected objects in the current frame. Consequently, they propose motion and appearance scores based on which the model associates between the existing tracks and the newly detected objects. In [BWL18], Bao *et al.* develop an algorithm using Markovian Random Fields for propagating the object masks to the nearby pixels in the future frames, utilizing the connectivity and neighborhood information for each pixel. Based on the assumption of foreground objects having a different motion pattern than the background, the authors in [Hu+18] compute the optical flow and drive the coarse object masks from the motion information. A refinement network then refines these initial masks into the expected resolution. However, the assumption made in this model can be largely violated in real-world scenarios. In a different approach, [Xia+18] utilizes optical flow to align the features from the past and the future frames, combining them to generate the object mask. They additionally compute a motion prior by applying distance transformation to the optical flow and employ it for enhancing the final object mask.

Matching-based Video Object Segmentation. Matching-based VOS methods estimate the object mask by comparing features extracted from each image to the object *template* features. The template features for the specific objects are obtained utilizing the segmentation mask provided for the first appearance. To this end, VideoMatch [HHS18] proposes a soft matching layer for computing the similarity between the image features to the foreground and background. FEELVOS [Voi+19] employs a local and global matching strategy in the semantic embedding space for better capturing the fine segmentation details, robustness to object appearance change, and modeling the long-range relations between objects, and [YWY20] improves this method by additionally incorporating the background information to attain a more robust and discriminative representation. In [Wug+18], a Siamese architecture is used to segment the object based on its similarity to the mask template in the first frame and the last predicted segmentation as extra guidance for the model. Similarly, [Yan+19] proposes a VOS model where the object mask at every time step is detected based on the similarity of the current frame to a set of assigned anchor frames. Following this idea, [Joh+19] suggests a generative approach for segmenting the target object, introducing an appearance module to learn the probabilistic model of the background and the foreground object. [Bha+20] extends the idea of learning an appearance model by utilizing meta-learning for training a parametric object appearance model. In [Che+18], the authors borrow ideas from metric learning and develop a training objective for effectively capturing the similarity between the image and the template features.

In STM [Oh+19], Oh *et al.* extend the template matching idea by employing an external memory to utilize the object appearance model over multiple frames and model the object appearance shift over time. [SHK20] improves STM by introducing Gaussian filtering into the architecture. The filtering mechanism serves as a location prior and assigns a higher matching weight to areas closer to the most recent object location. In [Hu+21], Hu *et al.* propose employing position encoding and modeling the object relations for improved segmentation consistency and robustness to distractors in the scene. [CTT21] suggests using squared Euclidean distance as the feature similarity function instead of the dot product used in [Oh+19]. To address the growing memory issue in these methods due to employing an external memory bank, [Li+22c] develop a fusion module that aggregates the spatiotemporal features based on relevance and importance hence keeping the memory size fixed. Similarly, [Lin+22] addresses this problem using the Expectation Maximization algorithm for object feature aggregation.

With the recent advances in transformers [Vas+17; Dos+20], recent works have attempted to use transformers for VOS. For example, [Duk+21; FS22] integrate variations of transformers to the VOS pipeline to benefit from the powerful non-local self-attention and cross-attention operations deployed in these models. In [YWY21], the authors propose a transformer-based association pipeline that allows for segmenting multiple objects simultaneously, considerably

reducing the inference time. They do so by projecting the object masks to the same space as the image features and concatenating them with the value vector in the attention module, in contrast to appending each object mask to the RGB image and then extracting the features. The follow-up work [YY22] further improves this method by decoupling the image features into two separate branches for modeling the object-specific and object-agnostic features.

6.3 RNNs for Video Object Segmentation

As the objective in VOS is to accurately track a set of target objects in a video, a logical solution would be to employ a memory component for memorizing the objects of interest in the scene. This way, the model gains the ability to store the relevant portions of the image that require tracking. However, training memory-based architectures such as RNNs requires a considerable amount of training data [Wug+18]. With the release of YouTubeVOS [Xu+18], the largest video object segmentation dataset to date, the authors demonstrated that having enough labeled data makes it possible to train a sequence-to-sequence (S2S) model for VOS.

In S2S [Xu+18], an encoder-decoder architecture is used following the architecture design in [BKC17]. Furthermore, an RNN layer is employed after the encoder (referred to as bottleneck) to track the object of interest in a temporally coherent manner. The choice of RNN as the memory module has several advantages. First, the RNN spatiotemporal features provide rich motion information and the object's location at each frame. Second, RNNs are more efficient compared to other types of external memory [Oh+19; SWF+15] and do not add considerable overhead to the model. We note that architectures based on external memory do not keep the order of events, and the memory size grows linearly with the length of the video.

The S2S model is illustrated with yellow blocks in Figure 6.1 where the segmentation mask is computed as (adapted from [Xu+18]):

$$h_0, c_0 = \text{Initializer}(x_0, y_0) \tag{6.1}$$

$$\tilde{x}_t = \text{Encoder}(x_t) \tag{6.2}$$

$$h_t, c_t = \text{RNN1}(\tilde{x}_t, h_{t-1}, c_{t-1}) \tag{6.3}$$

$$\hat{y}_t = \text{Decoder}(h_t) \tag{6.4}$$

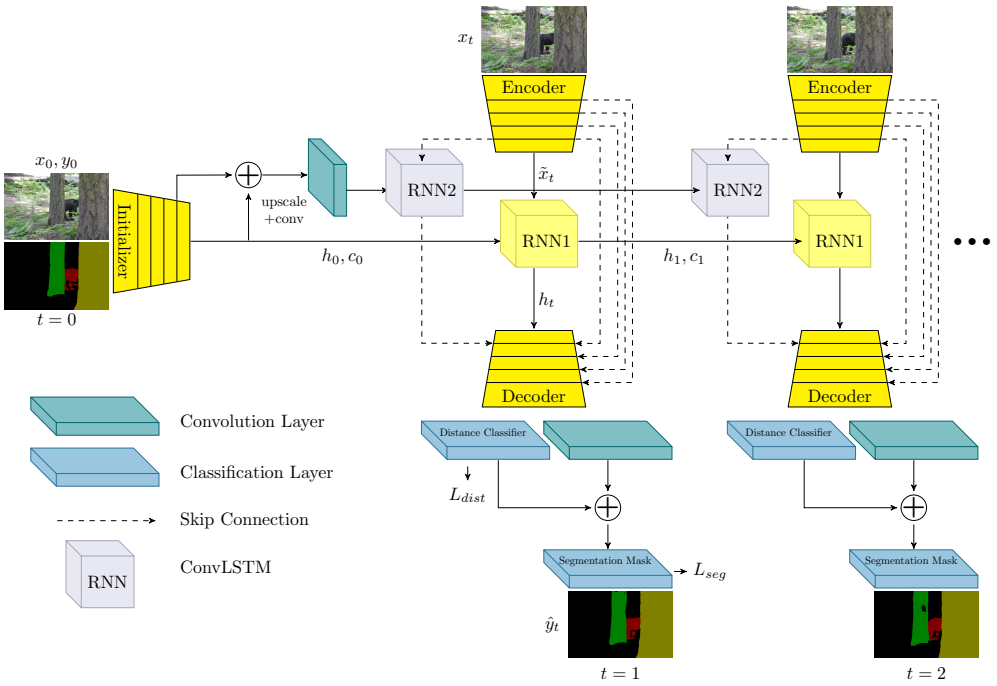


Fig. 6.1: The overall architecture of our approach. We utilize the information at different scales of the video by using skip-memory (RNN2). Experiments with multiple skip-memory connections are possible (only one is shown here for simplicity). We use an additional distance-based loss to improve the contour quality of the segmentation masks. For this purpose, a distance class is assigned to each pixel in the mask based on its distance to the object boundary. We use a $softmax$ at the distance classification branch and a $sigmoid$ at the segmentation branch to compute the L_{dist} and L_{seg}, respectively. Yellow blocks show the architecture of the original S2S model (Equations (6.1) to (6.4)), and all other blocks depict our extension to this model.

with x referring to the RGB image and y to the binary mask. In the next sections, we discuss our proposed solutions for improving the performance of the S2S model in tracking smaller objects and more accurate segmentation of the object boundaries.

6.3.1 Skip-Memory and Multi-Task Loss for RNN-based VOS

To better understand the role of the memory module used in the center of the encoder-decoder architecture in the S2S [Xu+18], we replaced the RNN layer with simply feeding the previous mask as guidance for predicting the next mask, similar to [Per+17]. Doing so, we observed a drastic performance drop of about ten percent in the overall segmentation accuracy. This suggests that only the guidance signal from the previous segmentation mask is not enough and that features from the previous time step should be aligned with the current time step. As a result, we hypothesize that the role of RNN in the architecture is twofold: First, remembering the object of interest through the recurrent connections and the

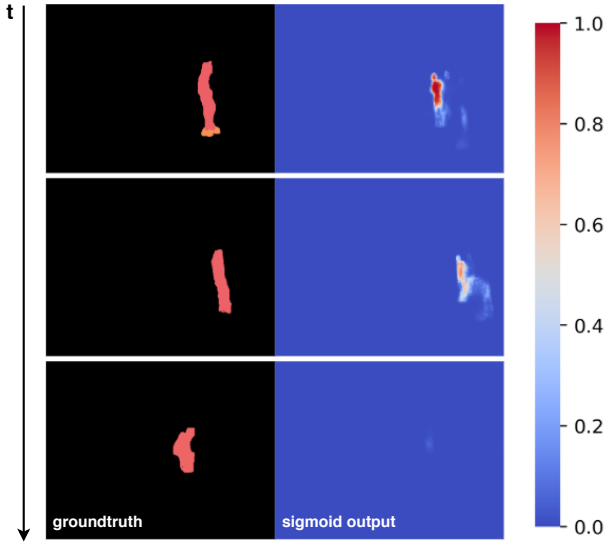


Fig. 6.2: In this image, the ground-truth mask is shown on the left, and the output of the decoder of the S2S architecture is on the right. The output of the $sigmoid$ function (last layer in the decoder) acts like a probability distribution over the binary classification, measuring the model confidence. The output of around 0.5 (white color coding) implies low confidence in the prediction, while values close to 0 or 1 (blue and red colors) show confident outputs w.r.t. to background and foreground classes). Our observation is that the model is not often confident when predicting masks for small objects. This uncertainty propagates to the following predictions causing the model to lose the target object within a few time steps. We argue that part of this issue is because the RNN located in the bottleneck of the encoder-decoder architecture does not receive enough information from the small objects. This leads to losing the object after a few time steps.

hidden state and masking out the rest of the scene. Second, to align the features from the previous step to the current step, having a role similar to optical flow.

Skip Memory

As mentioned earlier, the S2S model incorporates a memory module at the bottleneck of the encoder-decoder network to memorize the target object. By closely inspecting the output of this approach, we noticed that the predicted masks for small objects are often worse than the other objects (see Figure 6.4 and Figure 6.5 for visual examples). The issue is that the target object often gets lost early in the sequence, as shown in Figure 6.2. We reason that this is partially due to the lack of information for smaller objects in the bottleneck due to several pooling operations in the encoder network reducing the spatial resolution of the image features. For image segmentation, this issue is resolved by introducing skip connections between the encoder and the decoder [RFB15; BKC17]. This way, the information about small objects and fine details is directly passed to the decoder. Using skip connections is very effective in image segmentation; however, when working with video, if the information

in the bottleneck (input to the memory) is lost, the memory concludes that there is no object of interest in the scene anymore (since the memory provides information about the target object and its location). As a result, the information in the simple skip connections will not be very helpful in this failure mode.

As a solution, we propose a system that keeps track of features at different scales of the spatiotemporal data by using an RNN module in the skip connection as shown in Figure 6.1. In this work, we use Convolutional LSTM (ConvLSTM) [Xin+15] as the RNN module. We note that some technical considerations should be taken into account when employing ConvLSTM at higher image resolutions. As we move to higher resolutions (lower scales) in the video, the motion is larger, and also the receptive field of the memory is smaller. As stated in [Red+18], capturing the displacement is limited to the kernel size used in the ConvLSTM module. Therefore, adding ConvLSTMs at lower scales in the decoder without paying attention to this aspect might have a negative impact on the segmentation accuracy. Moreover, during our experiments, we observed that it is crucial to keep the simple skip connections (without ConvLSTM) intact in order to preserve the uninterrupted flow of the gradients. Therefore, we add the ConvLSTM in an additional skip connection (RNN2 in Figure 6.1) and merge the information from different branches using weighted averaging with learnable weights. Hence, it is possible for the network to access information from different branches in an optimal way.

For the training objective of the segmentation branch in Figure 6.1, we use the sum of the balanced binary-cross-entropy loss [Cae+17] over the sequence of length T, defined as:

$$L_{seg}(W) = \sum_{t=1}^{T}(-\beta \sum_{j \in Y_+} \log P(y_j = 1|X;W) - (1-\beta) \sum_{j \in Y_-} \log P(y_j = 0|X;W)) \quad (6.5)$$

where X is the model input, W is the learned weights, Y_+ and Y_- are foreground and background labeled pixels, $\beta = |Y_-|/|Y|$, and Y is the total number of pixels.

Multi-Task Loss: Border Classification

As the second contribution, we build upon the previous work of Bischke et al. [Bis+19] and train the network parameters in addition to the object segmentation mask with an image representation based on a distance transformation (see Figure 6.3 for an example). This image representation was successfully used in a multi-task learning setup to explicitly bias the model to focus more on those pixels which are close to the object boundary and more error-prone for misclassification compared to the ones further away from the edge of the object.

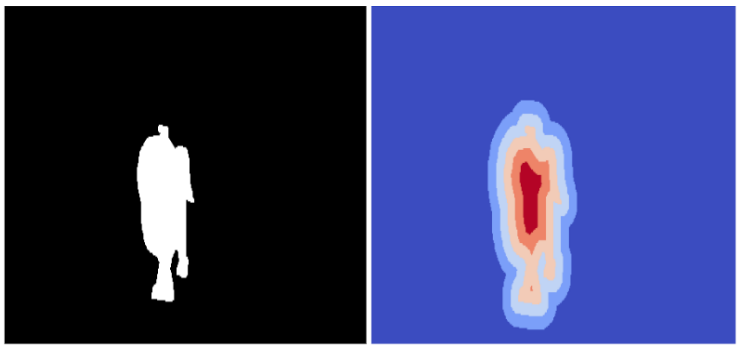


Fig. 6.3: In this figure, we show a binary mask (left) together with a heatmap depicting the distance classes (right) as explained in Equation 6.6. The number of distance classes is determined by two hyper-parameters for the *number of border pixels* around the edges and the *bin size* for each class. The visualization shows that, unlike previous works, our representations capture distance classes inside (reddish colors) as well as outside of the objects (blueish colors). The heatmap illustrates the distance classes with red for the inner class and blue for the outer class.

In order to derive this representation, we first apply the distance transform to the object segmentation mask. We truncate the distance at a given threshold to only incorporate the nearest pixels to the border. Let Q denote the set of pixels on the object boundary and C the set of pixels belonging to the object mask. For every pixel p we compute the truncated distance $D(p)$ as:

$$\begin{aligned} D(p) &= \delta_p \ \inf\{ \min_{\forall q \in Q} d(p,q), R \}, \\ \text{where } \delta_p &= \begin{cases} +1 & \text{if} \quad p \in C \\ -1 & \text{if} \quad p \notin C \end{cases} \end{aligned} \tag{6.6}$$

where $d(p,q)$ is the Euclidean distance between pixels p and q and R is the maximal radius (truncation threshold). The pixel distances are additionally weighted by the *sign function* δ_p to represent whether pixels lie inside or outside objects. The continuous distance values are then uniformly quantized with a bin-size s into $\lfloor \frac{R}{s} \rfloor$ bins. Considering both inside and outside border pixels, this yields $2 \times \frac{R}{s}$ binned distance classes as well as two classes for pixel distances that exceed the threshold R. We one-hot encode every pixel p of this image representation into k classification maps $D_K(p)$ corresponding to each of the border distance classes.

We optimize the parameters of the network with a multi-task objective by combining the loss for the segmentation mask L_{seg} and the loss for the border distance mask L_{dist} as a weighted sum as follows. Since we consider a multi-class classification problem for the distance prediction task, we use the cross-entropy loss. L_{dist} is defined as the cross entropy loss between the derived distance output representation $D_K(p)$ and the network output:

$$L_{total} = \lambda\, L_{seg} + (1 - \lambda)\, L_{dist} \tag{6.7}$$

The loss of the object segmentation task is the balanced binary-cross-entropy loss as defined in Equation 6.5. The network can be trained end-to-end.

6.3.2 Experimental Setup

In this section, we provide the implementation details followed by experimental results, a comparison with state-of-the-art methods, and qualitative results from our proposed method confirming the efficacy of our contributions in tracking smaller objects. We evaluate our method on the YouTubeVOS dataset [Xu+18], which is currently the largest dataset for video object segmentation. We use the standard evaluation metrics [Per+16b], reporting *Region Similarity* and *Contour Accuracy* ($J\&F$). J corresponds to the average intersection-over-union between the predicted segmentation masks and the ground truth, and F is defined as $F = 2\frac{precision \times recall}{precision + recall}$, regarding the boundary pixels after applying sufficient dilation to the object edges. For overall comparability, we use the *overall* metric of the dataset [Xu+18] that refers to the average of $J\&F$ scores.

Implementation Details

Initializer and Encoder Networks: The backbone of the initializer and the encoder networks in Figure 6.1 is a VGG16 [SZ14] pre-trained on ImageNet [KSH12]. The last layer of VGG is removed, and the fully-connected layers are adapted to a convolution layer to form a fully convolutional architecture as suggested in [LSD15]. The number of input channels for the initializer network is changed to 4, as it receives the RGB and the binary mask of the object as the input. The initializer network has two additional 1×1 convolution layers with 512 channels to generate the initial hidden and cell states (h_0, c_0) of the ConvLSTM at the bottleneck (RNN1 in Figure 6.1). For initializing the ConvLSTMs at higher scales, up-sampling followed by convolution layers are utilized in the same fashion as the decoder. Additional convolution layers are initialized with Xavier initialization [GB10].

RNNs: The ConvLSTM $1, 2$ (shown as RNN1 and RNN2 in Figure 6.1) both have a kernel size of 3×3 with 512 channels. The ConvLSTM at the next level has a kernel size of 5×5 with 256 channels. Here, we chose a bigger kernel size to account for capturing larger displacements at lower scales in the image pyramid obtained from the encoder ($\hat{x}$).

Decoder: The decoder consists of five up-sampling layers with bi-linear interpolation, each followed by a convolution layer with a kernel size of 5×5 and Xavier initialization [GB10]. The number of channels for the convolution layers are $512, 256, 128, 64, 64$, respectively. The features from the skip connections and the skip-memory are merged using a 1×1 convolution layer. To adapt the decoder for the multi-task loss, an additional convolution layer is used to map 64 channels to the number of distance classes. This layer is followed by a $softmax$ to generate the distance class probabilities. The distance scores are merged into the segmentation branch, where a $sigmoid$ layer is used to generate the binary segmentation masks.

Training Details: We use the Adam optimizer [KB14] with an initial learning rate of 10^{-5}. In our experiments, we set the value of λ in Equation 6.7 to 0.8. We choose this value by means of casual hyper-parameter tuning. When the training loss is stabilized, we decrease the learning rate by a factor of 0.99 every 4 epoch. Due to GPU memory limitations, we train our model with batch size 4 and a sequence length of 5 to 12 frames.

Border Output Representation: The number of border pixels and the bin size per class are hyper-parameters that determine the resulting number of distance classes. In our internal experiments (see Section 6.3.2), we noticed better results could be achieved if the number of distance classes is increased. In the following experiments, we set the *border_pixels*=20, the *bin_size*=1. Thereby we obtain for each object a segmentation mask with 42 distance classes (the number of output classes is $2 \times \frac{border\_pixels}{bin\_size} + 2$). Having the edge as the center, we have $\frac{border\_pixels}{bin\_size}$ classes at each of the inner and outer borders plus two additional classes for pixels that do not lie within the borders (inside and outside of the object) as shown in Figure 6.3.

Data Pre- and Post-Processing: In line with the previous work in multiple object video segmentation, we follow a training pipeline in which every object is tracked independently, and at the end, the binary masks from different objects are merged into a single mask. For pixels with overlapping predictions, the label from the object with the highest probability is taken into account. For data loading during the training phase, each batch consists of a single object from a different sequence. We noticed that processing multiple objects of the same sequence degrades the performance. The images and the masks are resized to 256×448 as suggested in [Xu+18]. For data augmentation, we use random horizontal flipping and affine transformations. We have not used any refinement step (e.g., CRF [KK11]) or inference-time augmentation. Moreover, we note that pre-training on image segmentation datasets can significantly improve the results due to the variety of present object categories in these datasets. However, in this work, we have solely relied on pre-trained weights from ImageNet [KSH12].

Method	Online training	J_{seen}	J_{unseen}	F_{seen}	F_{unseen}	overall
OSVOS [Cae+17]	Yes	59.8	**54.2**	60.5	**60.7**	**58.1**
MaskTrack [Per+17]	Yes	59.9	45.0	59.5	47.9	53.1
OnAVOS [VL17]	Yes	**60.1**	46.6	**62.7**	51.4	55.2
OSMN [Yan+18]	No	60.0	40.6	60.1	44.0	51.2
RVOS [Ven+19]	No	63.6	45.5	67.2	51.0	56.8
S2S [Xu+18]	No	66.7	48.2	65.5	50.3	57.7
S2S++(ours)	No	**68.68**	**48.89**	**72.03**	**54.42**	**61.0**

Tab. 6.1: Comparison of our best-obtained results with the state of the art approaches in video object segmentation using YoutubeVOS dataset. The values are reported in percentages and divided into columns for each score as in [Xu+18]. The table is divided to two parts for methods with and without online training. We can see that our approach (even without online training) achieves the best overall score.

Experimental Results

In Table 6.1, we provide a comparison between the results from our model and the state-of-the-art methods with and without online training. Online training is the process of further training at test time through applying a lot of data augmentation on the first mask to generate more data. This phase greatly improves the performance at the expense of slowing down the inference phase. As it can be seen in Table 6.1, the scores are measured for two categories of seen and unseen objects. This is a difference between other datasets and YoutubeVOS [Xu+18], which consists of new object categories in the validation set. Specifically, the validation set in the YoutubeVOS dataset includes 474 videos with 65 seen and 26 unseen categories. The score for unseen categories serves as a measure of the generalization of different models. As expected, the unseen object categories achieve a higher score when using online training (since the object is already seen by the network during the online training). However, despite not using online training (and therefore also having lower computational demands during test time), S2S and S2S++ achieve higher overall performance. It is worth mentioning that both F_{seen} and F_{unseen} scores improve by more than $4pp$ in our approach compared to the S2S model.

Figure 6.4 illustrates a qualitative comparison between our results and the ones from the S2S method. We provide additional examples in Figure 6.5. As can be seen from these examples, our proposed method can successfully track and segment smaller objects in the scene.

Ablations

Since the S2S method is the base of our work and the source code is not available, we provide a comparison between our implementation of S2S and our S2S++ in Table 6.2 when

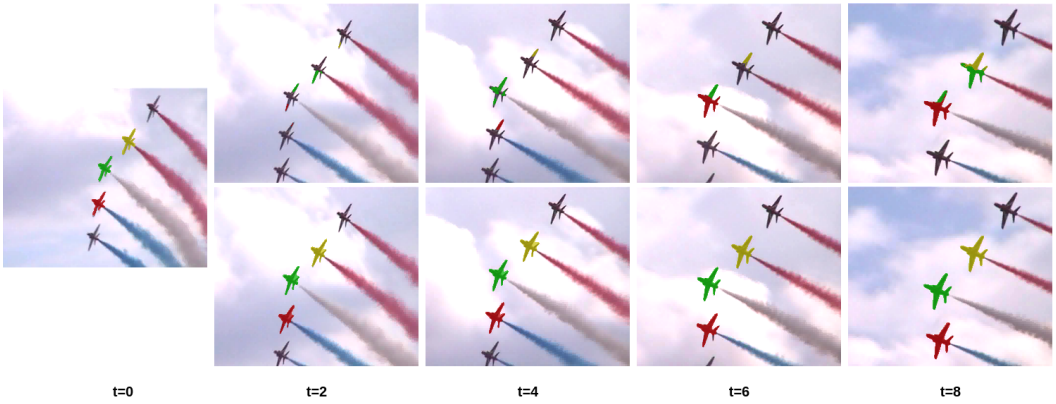


Fig. 6.4: Qualitative comparison between the results obtained from S2S approach (first row) and the results from our method (second row). The first mask ($t = 0$) is provided at test time and the target objects are segmented independently throughout the whole sequence. Every second frame is shown here and the brightness of the images is adjusted for better visibility. As it can be seen, our approach successfully tracks the target airplanes throughout the sequence while the S2S method loses and mixes the object masks early in the sequence.

Method	J_{seen}	J_{unseen}	F_{seen}	F_{unseen}	overall
S2S	65.4	43.6	67.9	47.5	56.1
S2S + multi-task loss	67.7	44.6	70.8	49.8	58.2
S2S + 1 skip-memory	66.9	46.8	69.2	50.1	58.3
S2S + 1 skip-memory + multi-task loss	67.2	47.0	70.2	52.3	59.2
S2S + 2 skip-memory + multi-task loss	**68.7**	**48.9**	**72.0**	**54.4**	**61.0**

Tab. 6.2: Ablation study on the impact of skip-memory and multi-task loss. We can notice that multi-task loss and skip-memory individually improve the results but lead to the best results when combined.

adding each component. As can be seen from the results, the best performance is achieved when using two skip-memory modules and multi-task loss. The additional skip-memory connections are incorporated at higher resolutions of the features computed by the encoder network.

We then take this model and experiment with different hyper-parameters for multi-task loss, as shown in Table 6.3. The results show that a higher number of border classes that is closer to regression yields a higher overall score.

It is worth mentioning that the distance loss (L_{dist}) has less impact on small objects, especially if the diameter of the object is below the border size (in this case, no extra distance classes will be added). Hence, we suspect the improvement in segmenting small objects (shown in Figure 6.4 and Figure 6.5) is mainly due to the use of skip-memory connections.

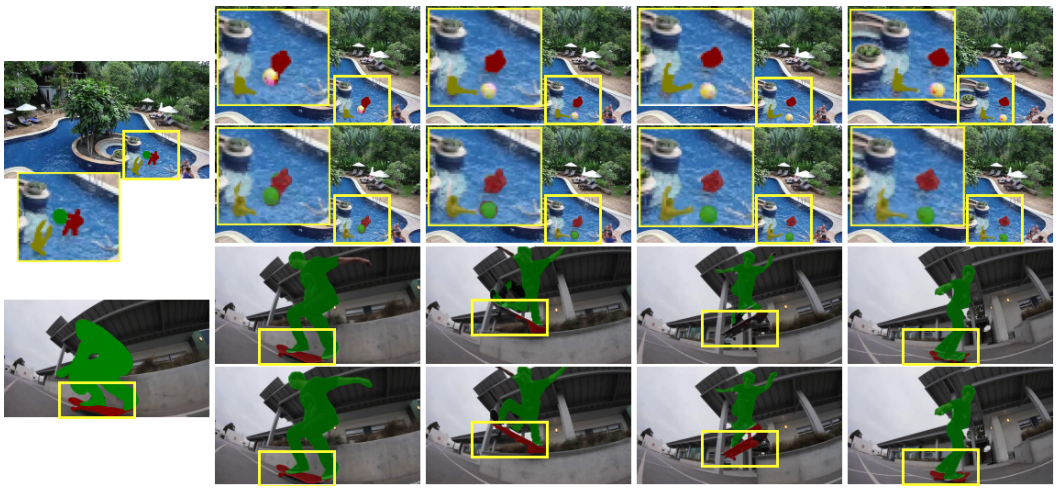


Fig. 6.5: Additional examples for qualitative comparison between the S2S method in the first row and ours in the second row. In the top part, the yellow frames were zoomed in for better visibility. We can observe a better capacity for tracking small objects.

border size	bin size	number of classes	J_{seen}	J_{unseen}	F_{seen}	F_{unseen}	overall
20	10	6	68.4	47.7	71.5	52.4	59.9
20	1	42	**68.9**	**48.8**	**72.0**	**54.4**	**61.0**
10	1	22	68.4	47.9	71.6	52.8	60.2

Tab. 6.3: Results for different hyper-parameters for the multi-task loss on our best model. We can see that a higher number of distance classes slightly improves the metrics.

6.3.3 Summary

In this work, we build on top of the S2S model, a sequence-to-sequence approach for video object segmentation. We empirically identify a limitation of this model for tracking small objects. To this end, propose using skip-memory connections for utilizing multi-scale spatiotemporal information of the video data. Moreover, we incorporate a distance-based multi-task loss to improve the predicted object masks for video object segmentation. In our experiments, we demonstrate that this approach outperforms state-of-the-art methods on the YouTubeVOS dataset [Xu+18]. Our extensions to the S2S model require minimal changes to the architecture and considerably improve the contour accuracy score (F) and the overall metric.

6.4 Correspondence Matching for Video Object Segmentation

Another approach proposed in the literature for solving VOS is through correspondence matching, that is, finding image features belonging to the object of interest compared to a set of reference features. The main idea is to estimate the object mask by computing the similarity between object features in the first frame (f_0) and image features at each time step (x_t), as the first mask of the object appearance is provided (note that $f_t \in x_t$). Accordingly, the image areas with the highest similarity to the reference object are marked as the current object features f_t.

One of the models proposed in the literature that builds on this idea is RGMP [Wug+18]. In this model, a Siamese architecture is employed with one encoder responsible for computing the reference object features f_{ref} and the second encoder for extracting the image features x_t. The computed features are then concatenated and passed to the matching block, which is implemented using a global convolution (GC) module. The GC module has a large receptive field; hence, it can adequately compute the similarity between features at different locations in the image. Having a sufficiently large receptive field is crucial as the features in the reference and current frames may not be located in the same neighborhood due to the object's movement. Finally, a decoder network consisting of multiple upsampling and refinement layers is employed to map the collected object features f_t to the final segmentation mask with the expected resolution. The overall architecture of this method can be seen in Figure 6.6

Pros and Cons of the Matching-based and RNN-based Methods

Matching-based methods work solely based on visual cues and similarity information; as a result, they fail under any circumstances where the object's appearance significantly changes. Moreover, similarity measures are not sufficient for distinguishing between similar objects.

RNN-based methods also suffer from several limitations. One of the main constraints of RNN-based models, such as S2S, is the fixed-sized memory, which can be insufficient to capture the whole sequence and long-term dependencies [BCB14]. Therefore, as the sequence length grows, access to information from earlier time steps decreases. This issue and the vanishing gradient problem adversely impact the segmentation quality in longer sequences, especially in videos with occlusion, where the target object can be absent for an extended period. Another obstacle with this category of approaches is drift and error

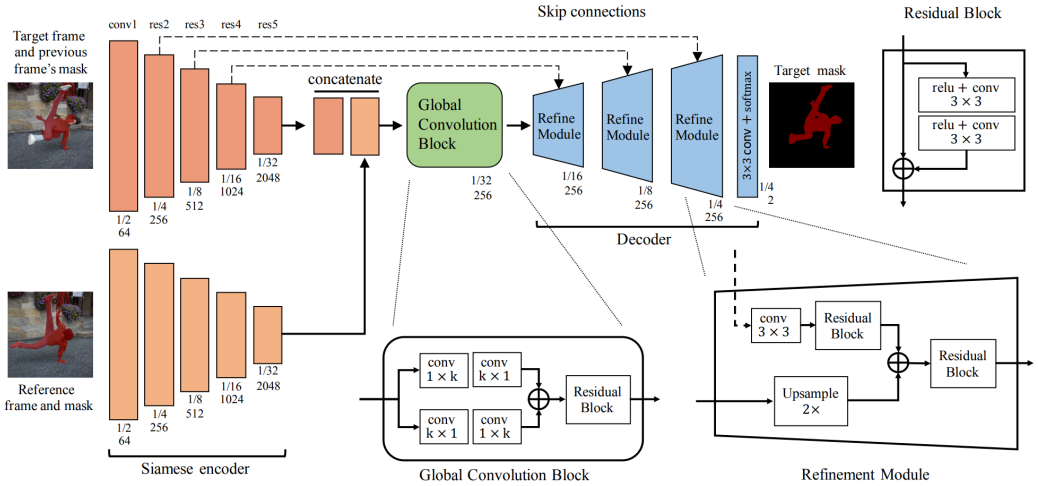


Fig. 6.6: In RGMP [Wug+18] method, a Siamese architecture is used for computing the reference object features as well as the image features. Consequently, a matching block computes the similarity between the image and the reference features. Finally, a decoder network maps the estimated object features to the segmentation mask at the expected resolution. Image source: [Wug+18].

propagation. Due to the recurrent connection, the model output is fed back to the network; thus, the prediction error propagates to the future, and erroneous model predictions affect the performance for future time steps. This issue is another contributing factor to the performance drop in later frames.

In the following, we propose an architecture that attempts to address the shortcomings of RNN- and matching-based methods by developing a hybrid architecture that combines the best of both worlds.

6.4.1 Feature Propagation and Matching Fusion for VOS

Based on the challenges in the matching- and RNN-based models, we propose Hybrid-S2S, a hybrid architecture combining the RNN output with information derived from correspondence matching. In our model, the segmentation mask is predicted using the location prior obtained from the RNN, as well as similarity matching between the video frames at $t-1$ and t. The merits of using the spatiotemporal model from RNN-based models and the matching-based methods are complementary. When multiple similar objects are present in the scene, the matching-based approaches struggle to distinguish between the different instances. Hence, the location prior provided by the spatiotemporal features from the RNN can resolve this ambiguity. Moreover, the information obtained from similarity matching provides a reliable measurement for propagating the segmentation mask to the next time step (as investigated in [Yan+19] for zero-shot VOS). Using this additional data

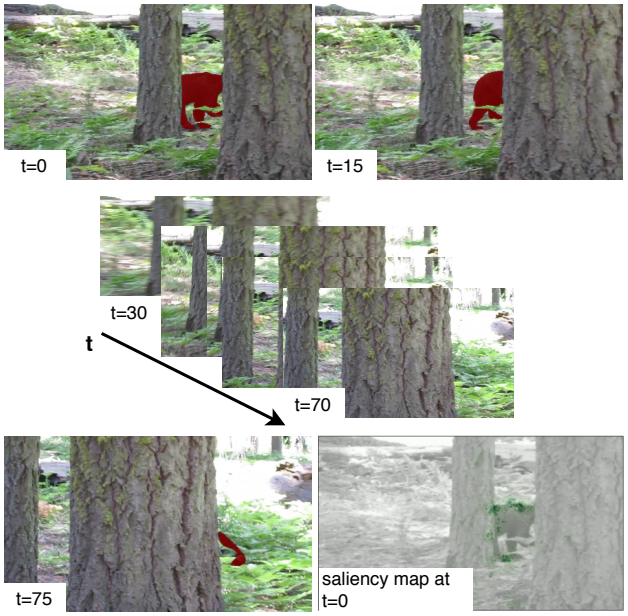


Fig. 6.7: This figure indicates how utilizing the first frame as the reference can help the model recover from occlusion. Here, the object of interest is a bear overlaid with the red mask, which is absent from the middle row frames (from $t = 30$ to $t = 70$). We observe that the model can detect the animal after it appears again, and by looking at the saliency map of the first frame, we note that the model has correctly captured the correspondence between the bear in the first frame and the frame right after the occlusion.

helps the model reduce the prediction error, improving the drift problem and obtaining better segmentation quality for longer sequences.

To encode the frame at $t-1$, we redefine the initializer network's task in S2S to a reference encoder (as shown in Figure 6.8), initializing the hidden states of the RNN module with $zeros$. In our experiments, we observed that the initializer network does not play a crucial role, and it is possible to replace it with zero-initialization with little change in the performance.

To perform the similarity matching between the RNN hidden state (h_t) and the reference encoder's output features, one can use different techniques such as using the cosine distance between the feature vectors. Here, we follow the design in [Wug+18] and use a Global Convolution [Pen+17] to accomplish the task (merge layer in Figure 6.8). Global Convolution (GC) approximates a large kernel convolution layer efficiently with less number of parameters. The large kernel size is essential to model both the local connections (as required for localization) and the dense global connections required for accurate classification (foreground, background). This way, the model directly accesses the features from time steps 0 and $t-1$. We note that this operation can also be interpreted as self-attention, as the

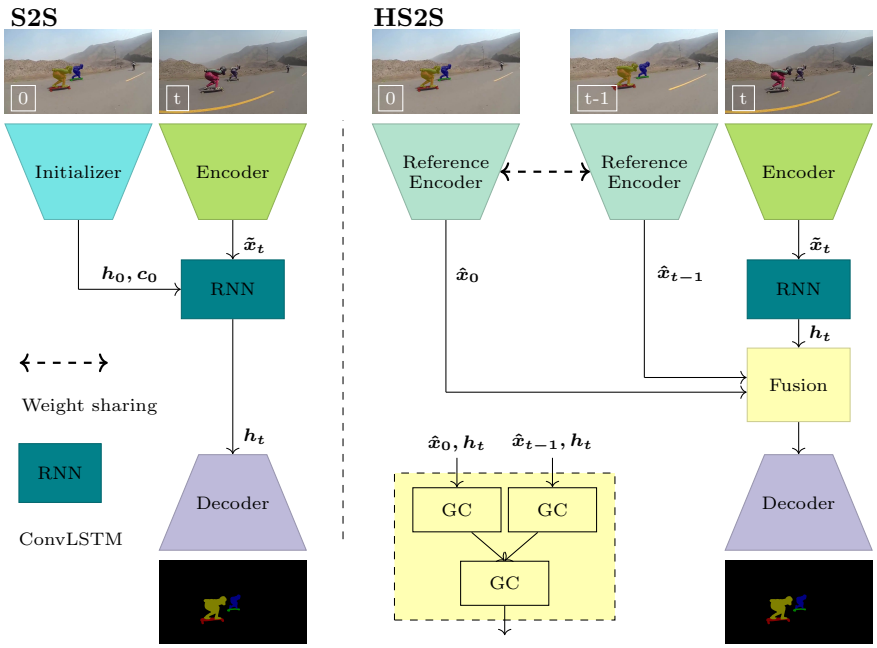


Fig. 6.8: In this figure, we depict the overall architecture of S2S [Xu+18] (Equations (6.1) to (6.4)) and our HS2S method (eqs. (6.8) to (6.13)). In HS2S, we initialize the RNN hidden states (h_0 and c_0) with *zeros*, instead of using the initializer network. We keep track of the target object by feeding the previous segmentation mask (y_{t-1}) to the encoder as an additional input channel, similar to [Per+17]. Furthermore, we use a separate reference encoder to process the input to the matching branch. We highlight that the functions approximated by these two encoders differ, as the inputs to the *Reference Encoder* are aligned in time, but this is not the case for the *Encoder* network. Finally, the hidden state of the RNN (h_t) is combined with the encoded features from the matching branch via a fusion layer and passed to the decoder to predict the segmentation mask. The skip connections between the encoder and the decoder networks are not shown for simplicity.

features at the current time step, which share a higher similarity to the object features from the reference frames, get a higher weight via the convolution operation in the merge layer.

As shown in Figure 6.8, we do not use weight sharing between the Reference Encoder and the Encoder, as we observed a considerable performance drop in doing so. We believe the underlying reason is that the functions approximated by these two modules are different; the input images to the Reference Encoder are aligned in time, while the inputs to the Encoder are not. We highlight that compared to S2S [Xu+18], the only added element is the lightweight Merge Layer (see Figure 6.8). The rest of the components remain unchanged by modifying the task of the Initializer Network to Reference Encoder.

Attention to the First Frame. As suggested in [Ebe+17] for the Video Prediction task, the first frame of the sequence is of significant importance as it contains the reference

information which can be utilized for recovering from occlusion. We note that by definition, the target object is present in the first frame. By computing the correspondences between the object appearance after occlusion and in the first frame, the model is able to re-detect the target. Additionally, [Yan+19; Wug+18] demonstrate the effectiveness of using the first frame as an anchor or reference frame. In [Wug+18], the authors propose a Siamese architecture that learns to segment the object of interest by finding the feature correspondences between the target object in the first frame and the current frame. Although this model's performance suffers in scenarios with drastic appearance changes, it reveals the importance of rigorously using the data in the first frame. We use the same reference encoder and merge layer for integrating the first frame features. We hypothesize that this modification can be considered as an attention mechanism [BCB14], where the attention span is limited to the first frame. Using attention is a standard solution to address this finite memory in the RNNs, by providing additional context to the memory module. The context vector is usually generated from a weighted combination of the embeddings from all the time steps. However, in high-dimensional data such as video, it would be computationally demanding to store the features and compute all the frames' attention weights.

The resulting architecture is shown in Figure 6.8 and can be formulated as:

$$h_0, c_0 = \mathbf{0} \tag{6.8}$$

$$\tilde{x}_0 = \text{Reference\_Encoder}(x_0, y_0) \tag{6.9}$$

$$\tilde{x}_{t-1} = \text{Reference\_Encoder}(x_{t-1}, y_{t-1}) \tag{6.10}$$

$$\tilde{x}_t = \text{Encoder}(x_t, y_{t-1}) \tag{6.11}$$

$$h_t, c_t = \text{RNN}(\tilde{x}_t, h_{t-1}, c_{t-1}) \tag{6.12}$$

$$\hat{y} = \text{Decoder}(\tilde{x}_0, \tilde{x}_{t-1}, h) \tag{6.13}$$

where x and y are the RGB image and the binary segmentation mask, and $\mathbf{0} \in R^d$ with d as the feature dimension. Here the merge layer is considered as part of the decoder. We train our proposed method end-to-end using the objective in Equation 6.7.

6.4.2 Experimental Setup

In this section, we present implementation details, quantitative and qualitative results, as well as an ablation study on the impact of different components in our HS2S method. Similar to the setup described in Section 6.3.2, we evaluate our model on the YouTubeVOS [Xu+18] dataset and report the *Region Similarity* and *Boundary Accuracy* ($F\&J$ scores) [Per+16b] for *seen* and *unseen* object categories. Moreover, we evaluate our method on DAVIS2017

dataset [Pon+17] without further finetuning to measure the generalizability of our proposed approach.

Implementation Details

Encoder Networks: In the original S2S model, a VGG network [SZ14] is used as the backbone for the initializer and encoder networks. In this work, we utilize a ResNet50 [He+16] architecture, pre-trained on ImageNet [Den+09]. We remove the last average pooling and the fully-connected layers, which are specific for image classification. Furthermore, we add an extra 1×1 convolution layer to reduce the number of output channels from 2048 to 1024. The number of input channels is altered to 4 as we feed the RGB image and the binary segmentation mask to the encoder. We utilize skip connections [RFB15] between the encoder and the decoder at every spatial resolution of the feature map (5 levels in total) to capture the fine details lost in the pooling operations and reducing the spatial size of the feature map. Moreover, we use an additional RNN module in the first skip connection, as suggested in [Azi+21a]. The impact of changing the backbone network in the S2S model from VGG to ResNet on the segmentation accuracy is studied in Table 6.8.

Fusion Layer: The role of the fusion module is to perform correspondence matching between the RNN hidden state (the spatiotemporal features) and the outputs from the reference encoder. There are different ways that can be used for this layer based on similarity matching and cosine distance. Similar to [Wug+18], we utilize Global Convolution (GC) layers [Pen+17] for this function. Two GC layers with an effective kernel size of 7×7 are employed to combine the RNN hidden state with the reference features and the features from the previous time step $\tilde{x}_0$ and $\tilde{x}_{t-1}$ as in Equations (6.9) and (6.10)). The outputs of these two layers are then merged using a 1×1 convolution and then fed into the decoder network. For the **RNN** layers and the **Decoder** network, we follow the same setup described in Section 6.3.2.

Training Details: For data augmentation, we apply random horizontal flipping as well as affine transformations. We use Adam optimizer [KB14] with an initial learning rate of 10^{-4}. We gradually lower the learning rate in the final phase of training when the loss is stable. During the training, we use video snippets with 5 to 10 frames and a batch size of 16. Additionally, we apply a curriculum learning method as suggested for sequence prediction tasks [Ben+15]. To this end, we use the ground-truth for the segmentation mask input in the earlier stages of training where the model output is not yet satisfactory. This phase is known as *teacher forcing*. Next, with a pre-defined probabilistic scheme [Ben+15], we randomly choose between using the ground-truth or the model-generated segmentation mask on a

Method	Online training	J_{seen}	J_{unseen}	F_{seen}	F_{unseen}	overall
OSVOS [Man+18]	Yes	59.8	54.2	60.5	60.7	58.8
MaskTrack [Per+17]	Yes	59.9	45.0	59.5	47.9	50.6
OnAVOS [VL17]	Yes	60.1	46.6	62.7	51.4	55.2
S2S(OL) [Xu+18]	Yes	**71.0**	**55.5**	**70.0**	**61.2**	**64.4**
OSMN [Yan+18]	No	60.0	40.6	60.1	44.0	51.2
RGMP [Wug+18]	No	59.5	45.2	-	-	53.8
RVOS [Ven+19]	No	63.6	45.5	67.2	51.0	56.8
S2S(no-OL) [Xu+18]	No	66.7	48.2	65.5	50.3	57.7
S2S++ [Azi+21a]	No	68.7	48.9	72.0	54.4	61.0
HS2S(ours)	No	**73.6**	**58.5**	**77.4**	**66.0**	**68.9**

Tab. 6.4: A comparison with the state-of-the-art methods on the Youtube-VOS dataset [Xu+18]. The upper part of the table shows models with online training and the lower part without. All scores are in percent. RVOS, S2S, and S2S++ are thN-based architectures. As shown in this table, our hybrid model outperforms the S2S(no-OL) baseline model with an average improvement of 11.2 pp.

per-frame basis. This process helps to close the gap between the training and inference data distributions (during the inference, only the model-generated masks are used).

Experimental Results

Table 6.4 shows a comparison of our model with other state-of-the-art methods. The upper and lower sections include the methods with and without online training. During online training, the model is further fine-tuned on the first frame (where the segmentation mask is available) at test time. Although this stage significantly improves the segmentation accuracy, it results in slow inference, which is not practical for real-time applications. Despite this, our model without online training still outperforms the S2S model with online training. The performance improvement compared to RGMP [Wug+18] (matching-based) and S2S [Xu+18] (RNN-based) models strongly indicates that both propagation and matching information are required for better segmentation quality. Moreover, our method achieves similar performance to STM [Oh+19] when training on the same amount of data (not using synthetic data generated from image segmentation datasets) without relying on an external memory module. Therefore, our model is less memory-constrained during the inference stage compared to methods using external memory that are prone to memory overflow for longer sequences (in [Oh+19], authors save every 5th frame to the memory to avoid the GPU memory overflow during the test phase).

Figures 6.9 and 6.10 illustrate visual examples from our model, also in comparison with the S2S baseline [Xu+18]. As we observe, our model can properly track the target object in the presence of similar object instances as well as occlusion.

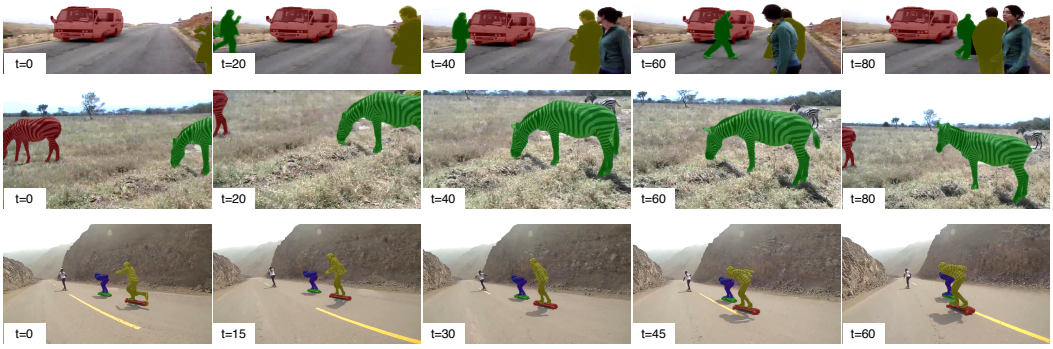


Fig. 6.9: Visual samples of our model on Youtube-VOS validation set. As can be observed, our method can successfully segment sequences with similar object instances, even in the presence of occlusion.

Method	J	F	$F\&J$
S2S [Xu+18]	-	-	-
RVOS [Ven+19]	52.7	58.1	55.4
RGMP [Wug+18]	58.1	61.5	59.8
HS2S- (ours)	**58.9**	**63.4**	**61.1**

Tab. 6.5: A comparison between the independent RNN-based (RVOS) and matching-based (RGMP) models and our hybrid method on the DAVIS2017 dataset [Pon+17] (test-val). HS2S- shows the results of our model trained on Youtube-VOS without fine-tuning on DAVIS2017. The results of the S2S model on DAVIS2017 were not available.

To assess the generalization of our model after training on Youtube-VOS, we freeze the weights and evaluate the model on DAVIS2017 dataset [Pon+17]. The results are provided in Table 6.5. We observe that our hybrid model outperforms the independent RNN-based and matching-based methods, even without fine-tuning on this dataset.

Ablations

Performance analysis for longer videos. To quantitatively assess our model's effectiveness, we evaluate it in challenging scenarios such as occlusion and longer sequences. As the validation set of the Youtube-VOS dataset is not released, we use the 80:20-splits of the training set from [Ven+19] for training and evaluation. For the S2S model results, we further used our re-implementation as the code for their work is not publicly available. Furthermore, we use the ResNet50 architecture as the backbone for both models for a fair comparison (to our disadvantage, as it improves the overall evaluation score of 57.3% for S2S (as reported in [Xu+18]) to 60% for our re-implementation S2S*).

Figure 6.11 shows the sequence length distribution of the Youtube-VOS training set (one sequence per object in each video). As can be seen, the length varies between 1 to about

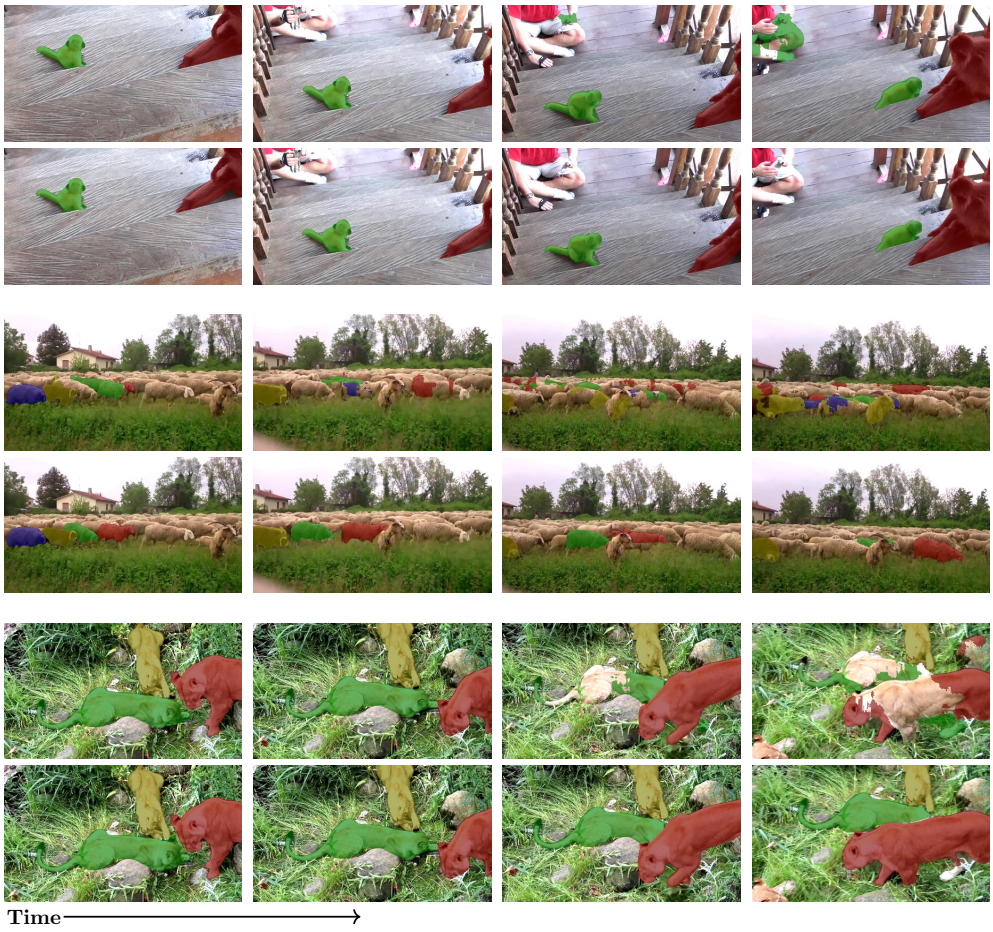


Fig. 6.10: Visual Comparison between the S2S and HS2S results in the upper and lower rows, respectively. We observe that our hybrid method can successfully maintain the segmentation accuracy at the later time steps.

35 frames in a very non-uniform fashion. To study the impact of the video length on the segmentation scores, we pick the sequences longer than 20 frames and measure the scores for frames with $t < 10$ (considered as early frames) and frames with $t > 20$ (considered as late frames) separately.

As presented in Table 6.6, we observe that the hybrid model improves the late frame accuracy significantly and reduces the performance gap between the early and late frames. This observation confirms the effectiveness of the hybrid path for utilizing the information from spatiotemporal features as well as correspondence matching.

Performance analysis for occluded videos. The histogram in Figure 6.12 shows the number of sequences with occlusion in YouTube-VOS training set. Each bin in the histogram shows

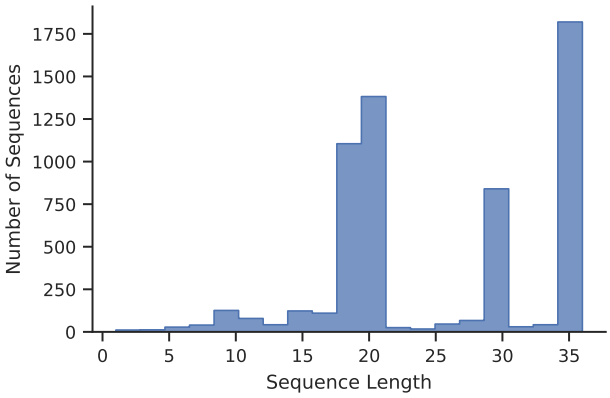


Fig. 6.11: Distribution of the sequence length (per object) in the Youtube-VOS dataset. In Youtube-VOS, the video frame rate is reduced to 30 fps, and the annotations are provided every fifth frame (6 fps). Therefore, a sequence with 36 labeled frames spans 180 time steps in the original frame rate.

Method	$F_{l<10}$	$J_{l<10}$	$F_{l>20}$	$J_{l>20}$
S2S*	74.4	73.7	54.5	54.6
HS2S(ours)	**77.1**	**76.3**	**65.5**	**64.2**

Tab. 6.6: A study on the impact of sequence length on segmentation accuracy. For this experiment, we picked video sequences with more than 20 frames. Then we compute the F and J scores for frames earlier ($t < 10$) and later ($t > 20$) in the sequence. As the results show, there is a performance drop as the time step increases. However, our hybrid model's performance drops a lot less than the baseline's.

the occlusion duration, and the y axis indicates the number of sequences that belong to each bin. As can be seen from this plot, the occlusion duration varies between 2 to 25 frames.

To study our model's effectiveness in handling occlusion, we report the scores for frames appearing *after* a first occlusion in Table 6.7. An occlusion is considered a scenario where the object leaves the scene entirely and re-appears again. As the areas below 100 pixels are almost not visible (and could be considered as labeling noise), we also consider occlusions at three different thresholds of 0, 50, and 100 pixels. As we can see in the table, occlusion is a challenging scenario with significantly lower scores than the average sequence scores.

Method	avg	$th:0$	$th:50$	$th:100$
S2S*	63.3	33.6	30.8	33.1
HS2S (ours)	**69.0**	**40.2**	**39.3**	**47.7**

Tab. 6.7: A study on the impact of occlusion on the segmentation quality. The scores presented in this table are the average of F and J scores in percentages, when considering different thresholds (in pixels) for occlusion. The avg score refers to the average result for all the sequences in the 20-split. For the other columns, we only considered the frames after ending the first occlusion period (when the target object re-appears in the scene).

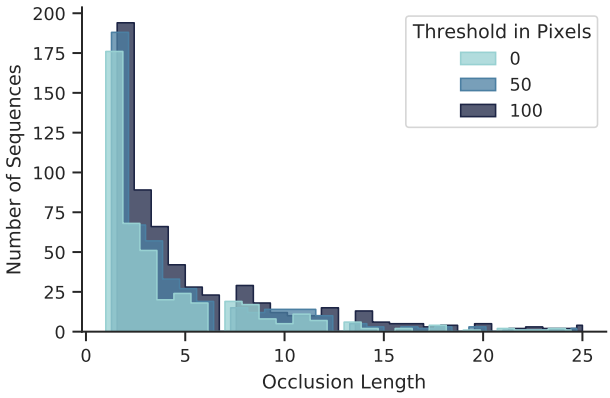


Fig. 6.12: The number of occluded sequences (per object) in YouTube-VOS train set for different occlusion lengths and with three occlusion thresholds (shifted by 1/3 for better visibility).

Method	J	F	$F\&J$
S2S [Xu+18]	57.5	57.9	57.7
S2S*	59.1	63.7	61.4
HS2S$_0$	64.0	68.95	66.5
HS2S$_{t-1}$	63.6	68.7	66.2
HS2S	**66.1**	**71.7**	**68.9**
HS2S$_{sim}$	64.35	69.35	66.9

Tab. 6.8: An ablation study on the impact of different components in our model. S2S* is our re-implementation of the S2S method with the same backbone as our model, for a fair comparison (this version achieves a better segmentation accuracy). S2S$_0$ refers to our model without the hybrid propagation, only using the first frame as reference. S2S$_{t-1}$ is our model with hybrid propagation and without utilizing the first frame. In HS2S$_{\text{sim}}$, we implemented the merge layer (Figure 6.8) using cosine similarity instead of Global Convolution.

However, our proposed approach again succeeds in defending its considerable improvement over the S2S baseline.

Ablation on model components and design choices. Table 6.8 presents the segmentation scores when different components in our model are added one at a time. The results for *S2S** are obtained from our re-implementation of the S2S model with ResNet50 backbone. As can be seen from the results, utilizing the first frame as the reference(HS2S$_0$) and using the hybrid match-propagate strategy(HS2S$_{t-1}$) both improve the segmentation quality. Moreover, the enhancements add up when they are integrated into a single model(HS2S). In addition, we provide the results for a variant of our model where we use cosine similarity [Wan+18] for the fusion layer instead of global convolution (referred to as HS2S$_{sim}$).

6.4.3 Summary

In this work, we present a hybrid architecture for the task of one-shot Video Object Segmentation. To this end, we combine the merits of RNN-based approaches and models based on correspondence matching. We show that the advantages of these two categories are complementary and can assist each other in challenging scenarios. Our experiments demonstrate that both mechanisms are required to obtain better segmentation quality. Furthermore, we provide an analysis of two challenging scenarios: occlusion and longer sequences. We observe that our hybrid model achieves a significant improvement in robustness compared to the baselines that rely on RNNs [Xu+18] and reference guidance [Wug+18]. However, occlusion remains an open challenge for future investigation, as the performance in this scenario is considerably lower than the average.

6.5 A Closer Look at What Matters in Hybrid VOS

In this section, we experiment with two architecture variants of the proposed HS2S model discussed in Section 6.4.1. In the first form, we explore the effectiveness of bidirectional design [SP97] where in addition to utilizing the information from the past time steps, we integrate the future frames via a bidirectional RNN network. In the second variation, we explore a multi-task training setup by joint training the VOS model together with the unsupervised optical flow objective. Our intuition is that since optical flow and VOS are well-aligned tasks (in both cases, the model has to learn pixel motion between the consecutive frames), training the model with both objectives might bring additional benefits via utilizing the optical flow-related constraints. Moreover, we perform extensive experiments and ablations on the YouTubeVOS dataset [Xu+18] to benchmark the role of various design choices in the HS2S model, including investigating different feature extraction backbones and memory modules.

6.5.1 Method I: Bidirectional Processing for VOS

In HS2S architecture, the video frames are processed sequentially, passing the information from the past to the future. Bidirectional sequence-to-sequence architectures enable the model to integrate information from the past as well as the future; they have been effective in improving the performance of sequential processing tasks such as Machine Translation [SP97; Sun+14; TAS17; GFS05]. Therefore, it is natural to conjecture that integrating the information from the future frames might benefit the HS2S model. However, based on the task definition in VOS, we need the object mask in the last frame ($t = T$) to process the video backward in time. Otherwise, the model will not recognize which object to track. To address this challenge, we design the bidirectional HS2S (Bi-HS2S) architecture shown in Figure 6.13.

As explained earlier, there are two different ways to inform the network about the object of interest. One is through using an initializer network that processes the first RGB and the mask frames and initializes the memory hidden states. The second way is by simply feeding the segmentation mask from the previous time step to the encoder network as a guidance signal. The second option does not fit the bidirectional design as in the backward processing of the video sequence, we do not have access to the initial object mask. As a result, we resort to the first alternative.

As illustrated in Figure 6.13, we initialize the memory hidden states with the initializer network in the forward path. For the backward processing, we simply initialize the backward memory with the last hidden state h_t obtained from the forward path. Our intuition is that

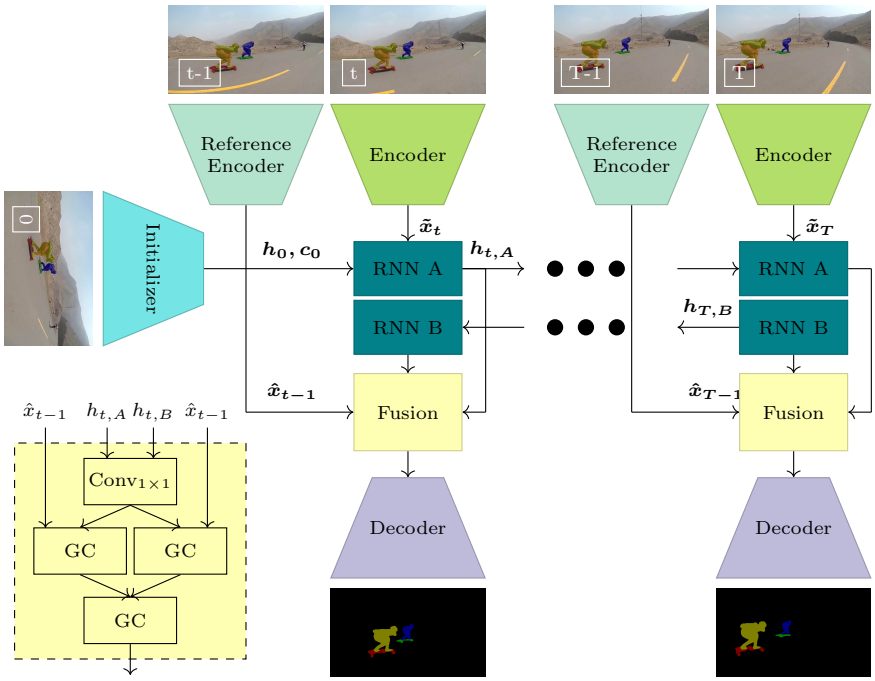


Fig. 6.13: The Bidirectional HS2S architecture. The hidden states from the forward and backward RNNs are combined using a convolution layer and then merged with the reference features and passed to the decoder.

h_t contains information about the target object at $t = T$ and can serve as a reasonable initialization. Finally, we combine the information from the forward and backward paths together with the reference features via the fusion block and pass it to the decoder to predict the segmentation masks.

6.5.2 Method II: Multi-Task Training with Optical Flow Objective

In multi-task learning, several tasks are combined within a single problem formulation and network architecture. This approach has been shown to be a successful training technique when the combined tasks are aligned in objective and can provide supplemental information to each other [Rud17]. In this section, we take inspiration from RAFT [TD20], a recent state-of-the-art optical flow architecture, and design an architecture that combines video object segmentation with optical flow prediction, referred to as RAFT-HS2S. Our intuition is that VOS and optical flow objectives are similar as they both tend to learn the pixel movement from one frame to the next. Accordingly, we explore whether combining these two learning objectives brings additional information to the model and enhances the segmentation accuracy.

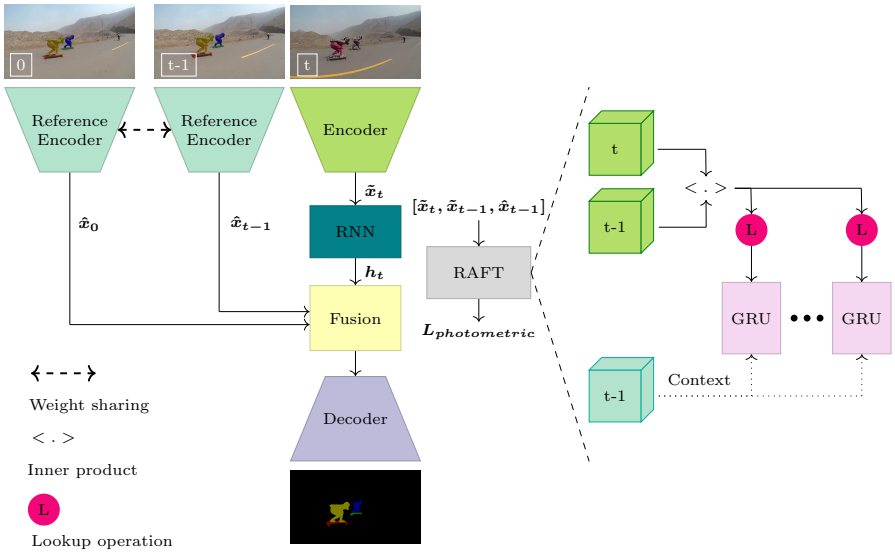


Fig. 6.14: The multi-task training setup in RAFT-HS2S, combining HS2S with an optical flow method named RAFT. RAFT module computes the correlation between frames at t and $t-1$ using the inner product between the respective feature vectors and generates an initial estimate of the optical flow between these consecutive frames. Then, it iteratively refines the approximated flow using a ConvGRU module that performs lookup operations based on the correlation volume and a context feature vector computed from the frame at $t-1$.

RAFT model [TD20] receives two consecutive images (x_t and x_{t-1}) as input and generates the flow field capturing the pixel motion between the consecutive frames. It consists of two encoders with the same architecture but separate weights; The first encoder extracts the features f_t and f_{t-1} while the second encoder only processes x_{t-1} to provide additional context to the network. Inspired by traditional optical methods, RAFT iteratively refines the estimated flow utilizing a ConvGRU [Su+20] that produces the flow delta at each time step.

Motivated by the commonality in VOS and optical flow training objectives, we adapt HS2S to accommodate the RAFT components as depicted in Figure 6.14. As can be seen in this plot, the reference encoder additionally takes the role of context encoding for the RAFT model, and the Encoder is employed for processing x_t and x_{t-1}. For the optical flow loss, we use an unsupervised objective, namely photometric loss [Jon+20]:

$$L_{photometric} = \sum |x^{(1)} - w(x^{(2)})| \tag{6.14}$$

Here $L_{photometric}$ is the photometric loss over all the pixels, and w is the warping operation that warps $x^{(2)}$ to $x^{(1)}$ using the optical flow between these two frames. This term implies

that having the precise motion, the pixel colors resulting from warping one image to the other should match.

6.5.3 Experimental Setup

In this section, we provide the experimental results for the proposed methods, followed by an extensive ablation on benchmarking the effect of various design choices on segmentation accuracy. The implementation details in this section follow the setup in Section 6.4.2 unless mentioned otherwise. We report the J and F scores on the YouTubeVOS dataset [Xu+18].

Experimental Results

In Table 6.9, we present the results obtained from bidirectional HS2S (Bi-HS2S) and optical flow multi-task training (RAFT-HS2S) compared to the HS2S baseline and the other state-of-the-art models. The results provided in the upper half of the table are for the methods with additional online training. Using the first object mask, these approaches further train the network at test time; as a result, they often achieve better accuracy but are considerably slower. As can be seen from the results in Table 6.9, we observe that utilizing the Bi-HS2S leads to further improvement of about 1pp while RAFT-HS2S achieves similar performance as HS2S. This implies that the information provided from the optical flow loss is already included in the VOS objective, and combining these additional terms does not bring additional benefits to the model.

Ablations

RNN Module. One of the main blocks in the HS2S architecture is the RNN block, which is accountable for memorizing the target object. In this section, we provide an ablation study of the HS2S performance when deploying three different RNN-based memories. The first variant is ConvLSTM [Xin+15]. This module is developed for processing sequential visual data by replacing the fully-connected layers in LSTM with convolution layers and adjusting the LSTM layer for visual pattern recognition.

As the second model, we study DeepRNN [Pan+19]. In this model, the authors address the challenges in training deep RNN models. Although deeper networks are expected to learn better representations compared to their shallow counterparts in the case of CNNs, deep RNN architectures designed by simply stacking the RNN layers do not lead to considerable improvement in the model accuracy. In [Pan+19], Pang *et al.* suggest this behavior roots in

Method	Online training	J_{seen}	J_{unseen}	F_{seen}	F_{unseen}	Overall
OSVOS [Man+18]	Yes	59.8	54.2	60.5	60.7	58.8
MaskTrack [Per+17]	Yes	59.9	45.0	59.5	47.9	50.6
OnAVOS [VL17]	Yes	60.1	46.6	62.7	51.4	55.2
S2S(OL) [Xu+18]	Yes	**71.0**	**55.5**	**70.0**	**61.2**	**64.4**
OSMN [Yan+18]	No	60.0	40.6	60.1	44.0	51.2
RGMP [Wug+18]	No	59.5	45.2	-	-	53.8
RVOS [Ven+19]	No	63.6	45.5	67.2	51.0	56.8
A-GAME [Joh+19]	No	66.9	61.2	-	-	66.1
S2S(no-OL) [Xu+18]	No	66.7	48.2	65.5	50.3	57.7
S2S++ [Azi+21a]	No	68.7	48.9	72.0	54.4	61.0
STM- [Oh+19]	No	67.1	63	69.4	71.6	68.2
TVOS [Zha+20b]	No	-	-	-	-	67.2
HS2S [Azi+21b]	No	73.6	58.5	77.4	66.0	68.9
Bi-HS2S (ours)	No	**74.9**	**59.6**	**78.0**	**66.7**	**69.8**
RAFT-HS2S (ours)	No	73.2	58.7	77.3	66.1	68.8

Tab. 6.9: Comparison of the experimental results on YouTubeVOS dataset [Xu+18]. The proposed bidirectional architecture (Bi-HS2S) leads to about 1pp improvement in the overall score, while multi-task training with optical flow estimation (RAFT-HS2S) does not improve the results.

the complex optimization operation when dealing with RNNs. Processing the entangled spatial and temporal information in sequential visual data leads to the optimization process becoming overly complex. This condition could become even more extreme for deeper RNNs, resulting in sub-optimal performance. To this end, they propose to disentangle the information related to the spatial flow from the temporal flow. They design a *Context bridge module (CBM)*, which is composed of two computing blocks for processing the representation and the temporal flows. By enforcing these sources of information to flow independently, the optimization process could potentially be simplified. In our experiments, we deployed a stack of 5 RNN layers following the setup proposed in [Pan+19].

In the third variant, TensorLSTM [Su+20], the authors attempt to improve the learning of long-term spatiotemporal correspondences for processing longer videos. They design a higher-order convolutional LSTM architecture named TensorLSTM that can better capture extended correlations. TensorLSTM consists of a preprocessing and a convolutional tensor-train module. The preprocessing module computes feature vectors from multiple overlapping sliding windows from the previous hidden states. These embeddings are then further processed through the convolutional tensor-train module and passed to the LSTM. Consequently, they are able to efficiently integrate the information from the previous hidden states and improve the capturing of long-term correlations.

As it can be seen from the results in Table 6.10, TensorLSTM achieves a better segmentation accuracy compared to the other variants. This implies that in HS2S architecture, we do not require deeper RNNs to carry the information about the object of interest. However,

Method	J_{seen}	J_{unseen}	F_{seen}	F_{unseen}	$overall$
ConvLSTM [Xin+15]	73.6	58.5	77.4	66.0	68.9
DeepRNN [Pan+19]	72.4	57.3	75.8	64.9	67.6
Tensor-TrainLSTM [Su+20]	**74.7**	**60.2**	**78.5**	**66.4**	**70.0**

Tab. 6.10: An ablation on the choice of RNN module. Employing Tensor-TrainLSTM [Su+20] results in improved segmentation performance.

accessing the information from multiple frames over an extended time period is beneficial for the model. In a way, TensorLSTM applies attention to a limited past context via the sliding-window mechanism in the preprocessing module. This observation is in line with employing the dual propagation strategy in Section 6.4.1 where simply merging the information from the time step $t-1$ improves the segmentation results.

Fusion Module. In this section, we study the model's performance when working with three different fusion block architectures in Table 6.11. Intuitively, the fusion block needs to provide global connections across the spatial dimensions as the object might be displaced to a further location compared to the reference frames. Additionally, it has to assign higher attention to the locations that belong to the foreground. In HS2S$_{sim}$, the spatiotemporal RNN features are merged with the reference features based on cosine similarity. HS2S$_{GC}$ merges these two branches using global convolution layers as suggested in [Pen+17] while HS2S$_{attn}$ replaces this operation with an attention layer [Ho+19]. We obtained similar performance for different fusion architecture options, but the design using attention attained the highest accuracy.

Method	J_{seen}	J_{unseen}	F_{seen}	F_{unseen}	$overall$
HS2S$_{GC}$	73.6	58.5	77.4	66	68.9
HS2S$_{sim}$	72.3	56.4	76.2	62.5	66.9
HS2S$_{attn}$	**73.9**	**58.7**	**77.5**	**66.3**	**69.1**

Tab. 6.11: Ablation on the impact of the fusion module combining RNN and matching information. We observe that attention-based fusion outperforms the other solutions, including using convolution layers with large kernel sizes.

Backbone Network. The encoder networks in Figure 6.8 are responsible for extracting descriptive features which will be then processed through the memory and decoded into a segmentation mask via the decoder network. Thus, improving the quality of the encoder network is expected to directly reflect on the segmentation quality. In this section, we study the behavior of HS2S model when employing various encoder architectures including VGG [SZ14], ResNet50 [He+16], DeepLab [Che+17b], and Axial-DeepLab [Wan+20b].

The DeepLab backbone is a modified ResNet50 with fewer pooling operations resulting in increased spatial feature dimension at the output (Higher spatial dimensions are presumably beneficial for dense prediction methods due to preserving fine local information).

Furthermore, it consists of an Atrous Spatial Pyramid Pooling (ASPP) module composed of a stack of convolution layers with various dilation rates. We experiment with a DeepLab backbone pretrained on image segmentation, with and without the ASPP module. Different than CNN-based backbones, Axial-DeepLab [Wan+20b] consists of fully-attentional blocks. In this model, the authors propose to break the attention into horizontal and vertical attentions to reduce the computational cost of the attention-based backbone (from quadratic to linear). However, this model still requires significantly higher memory compared to CNN-based backbones. Due to memory limitation, we experimented with *small* Axial-DeepLab architecture as elaborated in [Wan+20b].

As can be seen in Table 6.12, we obtained the best results when applying the ResNet-based encoder. Surprisingly, integrating the additional ASPP module from the DeepLab architecture resulted in lower performance. This behavior can be due to the added complexity resulting from combining the spatiotemporal RNN features with multi-scale processing in the ASPP module resulting in a more challenging optimization problem and sub-optimal performance.

6.5.4 Summary

In this work, we study two derived architectures building on our previously proposed HS2S model for solving VOS. Firstly, we introduce an architecture that enables bidirectional processing of video frames, effectively utilizing information from future frames. In the second design, we investigate the effectiveness of multi-task training by combining the VOS objective with optical flow estimation. This approach aims to leverage the additional information provided by optical flow to enhance the segmentation performance. Furthermore, we perform an extensive ablation study to quantify the impact of different components used in the HS2S model by investigating different designs for feature extraction backbones, RNNs, and fusion blocks used in the underlying architectures. These investigations reveal that our bi-directional extension (Bi-HS2S) improves over HS2S architectures by nearly 1 pp in the overall segmentation accuracy. To our surprise, our multi-task extension, also taking

Backbone	J_{seen}	J_{unseen}	F_{seen}	F_{unseen}	$overall$
VGG16 [SZ14]	71.4	56.0	74.9	64.8	66.8
ResNet-50 [He+16]	73.6	58.5	77.4	66.0	68.9
ResNet-101 [He+16]	73.9	58.5	77.3	66.2	69.0
ResNet-50-DeepLab (W/o ASPP) [Che+17b]	73.3	59.1	77.2	66.0	68.9
ResNet-50-DeepLab (W/ ASPP) [Che+17b]	72.3	56.9	76.5	64.2	67.4
Axial [Wan+20b]	70.5	55.0	73.1	61.8	65.1

Tab. 6.12: An ablation on the impact of the backbone network. We find that in our VOS pipeline, the ResNet architectures [He+16] perform better compared to other feature extractors.

optical flow into account (RAFT-HS2S), failed to improve over HS2S. In the expanded ablation study, we find that the ResNet-101-based backbone network, Tensor-TrainLSTM RNN architecture, and attention fusion blocks are the most beneficial design choices.

7 Self-supervised Learning for Spatiotemporal Correspondence Matching

7.1 Introduction

In this chapter, we focus on a key limitation of supervised learning in computer vision tasks: the reliance on labeled data. In this regard, we introduce two critical scenarios where the use of labeled data is especially problematic. The first scenario pertains to the high cost associated with annotating spatiotemporal correspondences in videos, where every object instance in each frame needs to be labeled. This intensive labeling process poses practical difficulties. The second scenario concerns the applicability of supervised methods when there is a mismatch between the data distribution during training and the inference; for instance, a situation where the training data is collected under sunny weather while the inference is conducted in foggy weather. This covariate shift can severely impact the model's performance. Although a trivial solution would be to fine-tune the model on the data from the test distribution, labeling data from each newly seen distribution is not feasible. Given these limitations, we ask the following question: Can we utilize self-supervised learning to alleviate the constraints imposed by supervised methods?

Within the last years, self-supervised learning methods have become highly popular due to their scalability and reducing the reliance on labeled data [Gui+23]. In particular, several works have introduced specialized pretext tasks for spatiotemporal correspondence learning, which involves learning feature correspondences over time and space [Jai+21; Von+18; JOE20; LLX20]. This is a fundamental problem in computer vision, as learning this objective enables the propagation of object features over time, benefiting various video applications such as video object segmentation, video instance segmentation, and video object tracking. However, the majority of the proposed approaches rely on color information as the supervisory signal, employing pretext tasks such as colorization [Von+18; LLX20]. Consequently, the model learns to find correspondences purely based on color information, resulting in a performance drop in crowded scenes with multiple similar objects. This limitation highlights the need to enhance the robustness of self-supervised methods further in challenging tracking scenarios.

Motivated by these observations, we delve into recent advances in self-supervised video correspondence learning in the first part of this chapter. Specifically, we concentrate on correspondence learning within the context of multiple object tracking (MOT). MOT is a difficult task due to the high similarity between object instances in crowded scenes, occlusion, and object appearance shift, with several applications in autonomous driving and robotics. To tackle MOT, we propose a self-supervised pipeline for frame-wise object correspondence/association learning that addresses several limitations of the current approaches under the demanding scenarios mentioned above. We introduce an appearance-based model that learns instance-aware object features tailored for identifying object correspondences. Our approach relies on two key components: optimal transport for soft differentiable matching, allowing for end-to-end training, and association pseudo labels derived from temporal and multi-view data using optical flow and disparity information. Unlike most self-supervised tracking methods that rely on pretext tasks for learning the feature correspondences, our approach is directly optimized for cross-object association in complex scenarios. As such, the proposed method offers a reidentification-based MOT approach that is robust to training hyperparameters and does not suffer from local minima, which is a challenge in self-supervised solutions.

In the second part of this chapter, we look into the issues raised due to the domain gap between the training and testing data, with a focus on video object segmentation and tracking, and seek potential remedies using self-supervised learning. In typical computer vision problems revolving around video data, pre-trained models are simply evaluated during the inference phase without any precautions to adapt the model to the test data distribution. This general approach clearly cannot capture the shifts that will likely arise between the distributions from which training and test data have been sampled [Wan+22]. Hence, adapting a pre-trained model to a new video encountered at test time could be essential to avoid the potentially catastrophic effects of such shifts. However, given the inherent impossibility of labeling data only available at test time, traditional fine-tuning techniques cannot be leveraged in this highly practical scenario. Encouraged by these challenges, we explore whether the recent progress in test time adaptation in the image domain and self-supervised learning can be leveraged to adapt a model to previously unseen and unlabelled videos presenting both mild (but arbitrary) and severe covariate shifts. Our experiments show that test time adaptation approaches applied to self-supervised methods are always beneficial. However, the extent of their effectiveness largely depends on the specific combination of the algorithms used for adaptation and self-supervision as well as the type of covariate shift taking place.

The content in this chapter is based on the publications [AMH23; Azi+22b].

7.2 Related Work

This section provides an overview of the literature related to studies conveyed in this chapter, including self-supervised learning in video data as well as its applications for domain adaptation.

7.2.1 Self-supervised Representation Learning

During the last few years, the AI community has witnessed immense success in self-supervised learning methods [JT20; LHS20]. These algorithms are of particular interest as the independence from the laborious data labeling process enables training on the massive amount of unlabeled data, alleviating the overfitting problem and improving generalization [TH21]. Earlier methods for representation learning in vision focused on designing pretext tasks such as rotation prediction [GSK18], solving jigsaw puzzles [NF16; Wei+19; Kim+18], permutation learning [San+18], image colorization [ZIE16; LMS17], and image context prediction [MHC18; DGE15]. The goal is to train the model to compute representative image features by teaching it to solve a carefully designed proxy task that encourages the model to understand meaningful visual patterns and contextual relations in an image.

Another popular line of work in self-supervised representation learning is contrastive learning [Che+20a]. Different than the aforementioned methods based on pretext tasks, contrastive learning trains the model by comparison [LHS20; He+20; Che+20b; Cui+21; OLV18]. In these methods, the model is trained to predict whether the provided input samples are similar based on a predefined similarity distance function. This way, the model is encouraged to pull together features from positive samples close and push apart the ones from negative samples to minimize the training objective. We note that the negative and positive data samples are assigned to either class without the need for any additional manual annotation.

However, the learned features by these approaches only perform well for recognition tasks but do not scale well to fine-grained feature correspondence learning [XW21]. The reason mainly lies within the proxy task design, conveying a relatively coarse objective (e.g., if multiple views belong to the same object or not). The following provides an overview of the algorithms specialized for self-supervised spatiotemporal learning.

Self-supervised Spatiotemporal Correspondence Learning. These methods design specialized proxy tasks to learn discriminative features essential for spatiotemporal correspondence learning from unlabeled video data [Von+18; JOE20; LX19a]. In the seminal work of Vondrick *et al.* [Von+18], the authors developed a self-supervised objective specialized to

video correspondence matching for the first time. They proposed learning feature similarities via video colorization, where they train a model to propagate color over video frames based on feature similarity. Accordingly, they utilize color information as the supervision signal. Follow-up works [LX19a; LLX20; WJE19] considerably improve the performance of this work by adding several enhancements such as cycle consistency, improved training procedure, using memory and attention mechanism to better represent the local and non-local relations between the objects present in a video clip. Differently, Jabri *et al.* [JOE20] formulate the problem of spatiotemporal correspondence learning as a contrastive random walk where the spatiotemporal feature propagation is learned via minimizing a cycle consistency training objective based on generating a palindrome from the video frames. [Bia+22] further improves this method by developing a hierarchical coarse-to-fine feature search over the temporal video frames. Alternative approaches formulate training objectives based on time cycle consistency [WJE19], or utilize motion information as the main training signal [Yan+21b]. In [Li+19a], Li *et al.* combine region-level correspondence matching (tracking) with colorization while the authors in [Yan+21a; KF19] utilize motion information for dense tracking.

In a different line of work, [XW21] successfully train a model for correspondence matching by simply employing a contrastive loss that determines whether two frames belong to the same video clip. In [Li+22b], the authors suggest enhancing the discriminative power of learned features by utilizing frames from *other* videos as additional negative examples and integrating the powerful location information using position encoding. [Jeo+21] develops an improved policy for positive and negative mining required for contrastive learning. To this end, the positive feature matches are selected based on a confidence measurement, and negative samples are incorporated with increasing difficulty levels as the training progresses. Furthermore, [Son22] reduces the adverse effect of noisy feature matches by introducing additional cycle consistency regularizations and considering the model uncertainty via a probabilistic model averaging algorithm. [Li+22d] addresses the limitation of self-supervised correspondence methods caused by purely utilizing pixel-level information and discarding the semantics. Orthogonal to these works, [Li+22a; Hon+22] focus on improving the architectural aspect and show that the simple ResNet-based feature extractor is not sufficiently robust for learning the feature representation and modeling the object relations in a video.

7.2.2 Object Correspondence Learning for Multi-object Tracking

In Multiple Object Tracking (MOT), the task is to track each detected object's bounding box throughout the object's appearance in the video. One of the most successful paradigms in MOT is *tracking by detection*. In this class of approaches, tracking is performed by

finding the frame-wise object correspondences to propagate the object ID (also referred to as the track ID). Classical methods that fall into this category rely on simple motion modeling [Bew+16; BES17] such as employing a Kalman filter. These methods build on the assumption that objects have a constant velocity – an assumption that is often violated in real-world scenarios. The recent OC_SORT method [Cao+22] makes several modifications to the Kalman-based formulation in [Bew+16], resulting in a considerable performance gain and better handling of occlusion.

With the progress of deep learning methods and significantly enhanced accuracy in object detection [Ren+15; Car+20], this improvement has naturally carried over to tracking by detection methods [BML19a]. Moreover, deep networks have enabled learning of better features which improves the association accuracy [WBP17; WB18]. Bergman et al. [BML19b] extend an object detector to a tracker using a regression network that estimates the object displacement, highlighting the importance of object detection in MOT pipelines. In [He+21], the authors improve on the standard frame-wise data association in MOT by modeling the intra-frame relationships between the tracks in the form of an undirected graph and formulating the problem as a graph-matching task. Similarly, [BL20; LGJ20; WKW21; Wan+21; Hyu+22] model the interaction between objects over multiple frames as a graph and utilize graph neural networks (GNN) to globally reason over object interactions and associations. In CenterTrack [ZKK20], a joint detection and tracking pipeline is developed for first detecting the object centers and then associating between them over consecutive frames via computing the distance between the object centers, taking the object motion offset into account. The follow-up work PermaTrack [Tok+21] utilizes the notion of physical object permanence and uses a recurrent module for memorizing the object track history and surmounting occlusion. [Tok+22] further improves this method by employing a consistency-based objective for object localization in occluded videos. Motivated by the success of transformer-based methods in vision applications [Vas+17; Dos+20], transformers have also been deployed in tracking algorithms [Zen+22; Sun+20b; Mei+22; Cai+22] to allow for better modeling of object relations. Sun et al. [Sun+20b] were the first to suggest a transformer-based architecture for learning the object and track queries used for detecting objects in succeeding frames and performing association. Trackformer [Mei+22] proposes tracking by attention, a model which uses a transformer encoder and decoder to perform the task of set prediction between the object detections and the tracks in an autoregressive manner. MeMOT [Cai+22] additionally utilizes an external memory for modeling the temporal context information. In MOTR [Zen+22], Zeng et al. extend the deformable DETR [Zhu+20] by building on the idea of object-to-track and joint modeling of appearance and motion by introducing a query interaction module and temporal context aggregation. Although reaching accurate tracking results, these methods rely on video-level tracking labels.

Differently, Bastani *et al.* [BHM21] develop a self-supervised algorithm to learn the object associates over multiple frames using a cross-input consistency objective that encourages the same association via visual and motion information. Although this method achieves good tracking results in high frame rate videos with minimal object motion, the performance degrades in low frame rate setup and large object motion due to the inherent dependency on motion smoothness assumption.

7.2.3 Domain Adaptation

Domain adaptation aims at adjusting a model trained on one domain to perform well on new and unseen domains by addressing differences in characteristics such as image appearance or lighting conditions. Based on the availability of labeled or unlabeled data from the target domain (test data) during the training, domain adaptation techniques can be divided into multiple sub-categories, as will be discussed in the following.

Unsupervised Domain Adaptation addresses a setup where the labeled data from a source domain and unlabeled data from a target domain are available during the training phase. The goal is to maximize the performance on the target domain [WC20]. In this regard, [SFS17; HHK18; Kum+18] propose feature alignment and adjusting the statistics of the source and target data distributions by applying linear transformations on the source features to lessen the impact of domain shift. Carlucci *et al.* [Car+17] develop domain alignment layers that apply domain-specific operations and align the features from the source and target distributions to a reference and can be embedded in any network. Similarly, [Pen+19] introduces a moment-matching component for multi-source domain adaptation, which is responsible for adapting the input domains to a target distribution. [Tse+20] employs a feature-wise transformation layer that learns the parameters of a linear operation used for modulating the activations towards adapting them to the target task/domain. In a different setup, Liang *et al.* [LHF20] suggest an effective transfer learning approach in a scenario where a pre-trained model is to be adapted to a target domain without having access to the source data.

Domain Generalization considers a more general scenario where the target data distribution is unavailable during training [Zho+21]. The goal is to improve the performance on the target domain with a focus on enhancing the *training* process. In this respect, [Car+19] proposes a multi-task setup and shows that training together with the auxiliary task of solving the jigsaw puzzle [NF16] improves the generalization to unseen domains. [Li+19b] proposes a meta-learning approach in which the objective for improving the generalization is learned itself, in contrast with methods that utilize manually designed loss functions [MBS13; BSC18]. Several works have studied this aspect in an attempt to accustom the

normalization layer to the target distribution [Li+16; Nad+20; Sch+20; Seo+20; Zha+20a; BS21]. For example, [Zha+20a] develops a domain-invariant normalization layer for stereo-matching by normalizing the features along the spatial and the channel dimensions to enforce the domain invariance in the learned representation while [Seo+20] proposes a domain-specific normalization layer for multi-source domain generalization by combining batch normalization [IS15] with instance normalization [UVL16] where the combination weights are learnable parameters of the network.

Test Time Adaptation, unlike the previously mentioned methods, performs domain generalization by learning from the data available at *test* time. In this respect, Sun *et al.* [Sun+20c] propose a multi-task setup using supervised and self-supervised objectives where the auxiliary loss is used to further fine-tune the network during inference. In [Wan+20a], the authors utilize entropy minimization [GB+05; Shu+18; Sai+19; Roy+19] to modify the modulation parameters of the BN layer to mitigate the impact of covariate shift between the training and testing data distributions and [Nad+20; Sch+20; Li+16] suggest updating the normalization statistics of the BN layer as an effective way of adapting the features to the target domain.

7.3 Self-supervised Learning for Video Object Correspondences

Multiple object tracking (MOT) is a fundamental task in computer vision with applications across domains, including scene understanding and autonomous driving. In many of these applications, tracking plays a safety-critical role for downstream planning and control algorithms. Accurate MOT requires precise detection of one or multiple object categories and correctly associating them throughout object presence in a dynamic scene. This task is challenging not only due to the similarity of object instances in the scene and highly dynamic object motion paths but also the fundamental problem of partial and full occlusions, which from the observer's view, can break object paths into separate segments.

Although a large body of work has explored MOT during the last years approaching the task from various viewpoints [WSH22; Rak+21; Cia+20; LKR22; He+21; Guo+21], the majority of these methods are supervised and rely on a highly laborious data labeling process. This limits the datasets that can be used to small ones like KITTI [GLU12], which only has 21 training sequences. As such, this often restricts the potential of tracking architectures, as more data can significantly improve the performance of deep learning-based methods [Tok+21]. In this work, we aim to utilize the large amount of unlabeled video data for MOT by utilizing the recent advances in object detection models, as elaborated in the following.

Fig. 7.1: We propose S^{3}Track, a self-supervised method for learning the object associations throughout a video by learning a robust appearance model. We use optimal transport for computing the soft object assignments, enabling end-to-end training of our model with association pseudo-labels. Our method shows strong performance in challenging scenarios such as occlusion and fast motion in the top row, severe weather conditions, and appearance change in the bottom row (see the objects pointed at with the white arrow). The track IDs are visualized by the bounding box color and the number inside. Data samples are from the nuScenes dataset [Cae+20] validation split using the provided detection bounding boxes.

Object detection is closely related to MOT, as one of the main strategies for solving MOT is *Tracking by Detection* which is learning to associate between the detected objects [Bre+09]. Object detection methods are trained on separate still images; thus, the labeling process is significantly simpler than MOT, which requires annotating sequences. Furthermore, the field has produced very accurate object detectors [Jia+19; Zai+22] that can be used to generate the detections for MOT. Although recent object detection practices utilizing transformers [Car+20; Liu+21] show promising performance, two-stage detection methods relying on region proposals [Ren+15] are still faring among the best-performing models in a wide range of detection tasks, thanks to employing techniques such as Region of Interest (RoI) pooling and hierarchical feature processing that has been proven crucial for object detection [He+15; Lin+17]. Nevertheless, it is an open question if we can rely on the accuracy of these existing mature detection models for MOT.

In our work, we assume access to a trained object detector for generating the detection bounding boxes and train an MOT model *without using video-level association labels*. With per-frame detections in hand, we propose a method to obtain association pseudo-labels using motion information over short video sequences. Using the detections and the RoI pooling layer, we extract the object features and compute an affinity matrix between the detections in the source and target frames. We propose *differentiable* optimal transport for finding the soft assignments between the detections, facilitating end-to-end training of our model using the association pseudo-labels. Thanks to this differentiable training, our model is able to compute robust and discriminative features optimal for object association, as shown in Figures 7.1 and 7.2. We validate the method on multiple tracking benchmark

Fig. 7.2: Heatmaps **a** and **b** show the cosine distance between object embeddings from an instance-agnostic model trained for object detection and our model trained for object association using optimal transport soft assignment at frames t_0 and t_1. The soft assignment mechanism is essential for obtaining instance-aware discriminative object features. Without this, features from different objects are not well separated in the embedding space, resulting in a low distance between multiple object instances and false matches (red in heatmap **a** shows the false associations: cars 2, 5, 6, 7 at t_1). Note that our method correctly matches all detections and adequately initializes a new track ID here for car 6 entering at t_1. (*: unmatched detection resulting in a new track ID).

datasets, including KITTI [GLU12], Argo [Wil+21], nuScenes [Cae+20], and Waymo datasets [Sun+20a], outperforming all tested unsupervised MOT methods.

The remaining of this section discusses the components of the proposed MOT pipeline, including the architecture, the differentiable object assignment using optimal transport, and the association pseudo-label generation. This is followed by discussing the implementation details and presenting the experimental results.

7.3.1 S^{3}Track: Self-supervised Tracking with Soft Assignment Flow

In conventional supervised MOT, a model is trained to predict a unique track ID for each object by minimizing a classification loss. In our setting, we do not have access to the track IDs; instead, we approach MOT as finding the frame-wise association between detected objects in the reference and target frames. Our goal is to learn an affinity matrix measuring the distance between detected objects in a reference (I_r) and a target video image (I_t).

Considering these detections, we predict the unique correspondences between the objects such that objects with the closest distance are matched.

At first glance, this resembles the common inference strategy in MOT approaches that formulate a distance matrix based on motion or appearance information and use the Hungarian algorithm to find the unique assignments. However, the bipartite matching via the Hungarian algorithm is *non-differentiable* and hence, does not facilitate learning correspondences. To tackle this challenge, we find soft associations between the reference and target objects by posing association as an optimal transport [Cut13; Mun57] problem. Having defined a *differentiable matching step*, we learn feature embeddings optimal for matching in an end-to-end training approach. We find this differentiable matching step essential for the proposed method; see Figure 7.2. We train our model with a negative log-likelihood objective where the ground truth association labels are replaced with assignment pseudo-labels obtained from video motion information.

In summary, our algorithm is split into the following steps:

- Compute the object features using the detection bounding boxes in a reference and target frame.
- Compute the distance matrix between the object features and find the object assignments using optimal transport.
- Compute the association pseudo-labels based on object overlaps by first aligning the detections using motion information (optical flow and disparity), then computing the distance matrix using object IoUs, and finally applying the Hungarian algorithm for obtaining the object assignments with the highest area overlap.
- Use the pseudo labels to train the model with a metric loss for learning the right correspondences using visual similarity information.

The overall architecture is illustrated in Figure 7.3. In the following, we discuss all components of the proposed method.

Optimal Transport for Soft Object Association

We solve the task of finding the corresponding objects in I_r and I_t with minimal assignment distance using optimal transport [Cut13; Mun57]. We will see that this approach allows for a fully differentiable matching process. Consider two discrete distributions a and b and a matrix C, which represents the cost of transporting distribution a to b using a probability

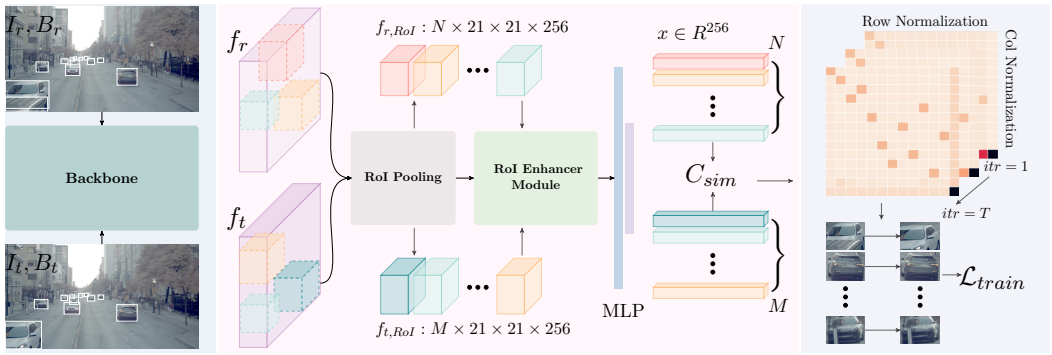


Fig. 7.3: The S^{3}Track architecture. We use a feature pyramid pooling network as the backbone and an RoI pooling layer for extracting the per-object features f_{RoI}, followed by an RoI Enhancer Module generating the instance-specific discriminative representation for each object. We compute the final embeddings x_i using an MLP and find the soft assignments between the embeddings using the differentiable optimal transport layer.

matrix P (or transport matrix). Optimal transport is a linear assignment algorithm [Cut13] that finds the P which minimizes the overall transport cost, that is

$$d_C(a, b) \coloneqq \min_{P \in U(a,b)} \langle P, C \rangle, \tag{7.1}$$

where $U(a, b)$ is the set of possible transport strategies

$$U(a, b) \coloneqq \{P \in R_+ \mid P\mathbf{1} = a, P^T\mathbf{1} = b\}. \tag{7.2}$$

We are interested in finding the assignments between the detections in I_r and I_t such that the objects with the highest similarity (lowest distance) are matched. In our work, we learn features optimal for object association. Consider $X_1 \coloneqq \{x_{1,1}, x_{1,2}, \ldots, x_{1,n_1}\}$, $X_2 \coloneqq \{x_{2,1}, x_{2,2}, \ldots, x_{2,n_2}\}$ as the set of extracted detection embeddings from the reference and target frames, where $x_{i,j} \in R^{256}$. We define the following cost matrix for feature similarity

$$C_{sim,ij} = 1 - \left\langle \frac{x_{1,i}}{\|x_{1,i}\|}, \frac{x_{2,j}}{\|x_{2,j}\|} \right\rangle, \tag{7.3}$$

where $\langle ., . \rangle$ represents the Frobenius dot product.

Adding an entropy regularization term to Equation (7.1) [Cut13] turns the task into a convex optimization problem that can efficiently be solved using the Sinkhorn algorithm [SK67]. This algorithm consists of differentiable operations, namely, iterative normalization of the rows and columns of matrix C until convergence (or with a fixed number of iterations), as shown in Figure 7.3. With this in hand, Equation (7.3) allows us to *learn the feature embeddings optimal for matching*.

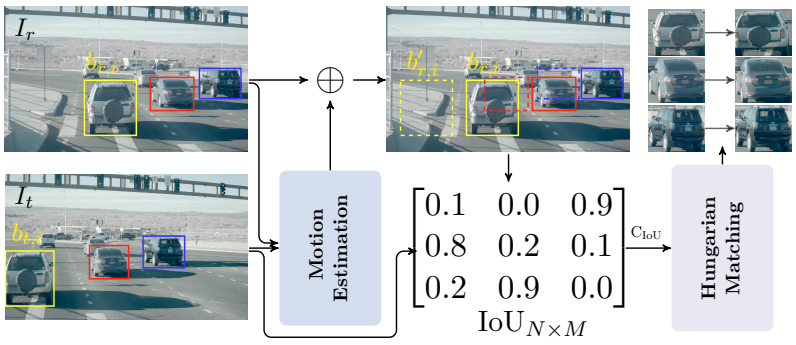


Fig. 7.4: Object association pseudo-label generation process. We align the detection bounding boxes using motion information between a reference and a target frame. We compute a cost matrix based on the IoU between the aligned objects and employ the Hungarian algorithm to find the corresponding objects with maximum bounding box overlap.

Flow-based Pseudo-label Generation

For training our association model, we recover pseudo-labels from temporal cues in video sequences and multi-view cues in stereo captures as shown in Figure 7.4. To this end, we perform motion compensation to align the object bounding boxes in the reference and target frames. We first estimate the motion between the reference and target frames I_r and I_t. Assume $B_r := \{b_{r,1}, b_{r,2}, \ldots, b_{r,n_1}\}$ and $B_t := \{b_{t,1}, b_{t,2}, \ldots, b_{t,n_2}\}$ are the detection bounding boxes in I_r and I_t. Then we align the detections from the reference to the target by forward warping

$$b'_{r,i} = b_{r,i} + M^t_{b_i,r}, \tag{7.4}$$

where $M^t_{b_i,r}$ is the computed motion vector at the center of detection box $b_{r,i}$. In the next step, we assign pseudo-labels based on the Intersection over Union (IoU) between the motion-adjusted object bounding boxes. Assuming aligned bounding boxes b_t and b'_r, we match objects with the highest overlap with the distance matrix

$$C^{\text{IoU}}_{ij} = 1 - \text{IoU}(b'_{r,i}, b_{t,j}). \tag{7.5}$$

We compute the unique and hard object association labels (i, j) using the Hungarian algorithm, which gives us object correspondences with the highest overlap (minimum cost). Note that, at this stage, we can employ the Hungarian algorithm since we do not require differentiability in the pseudo-label generation. When using temporal data, the I_r and I_t are temporally spaced video frames, and motion M is estimated using optical flow. When using stereo data, I_r and I_t are the left and right images, and M is the disparity between the two views.

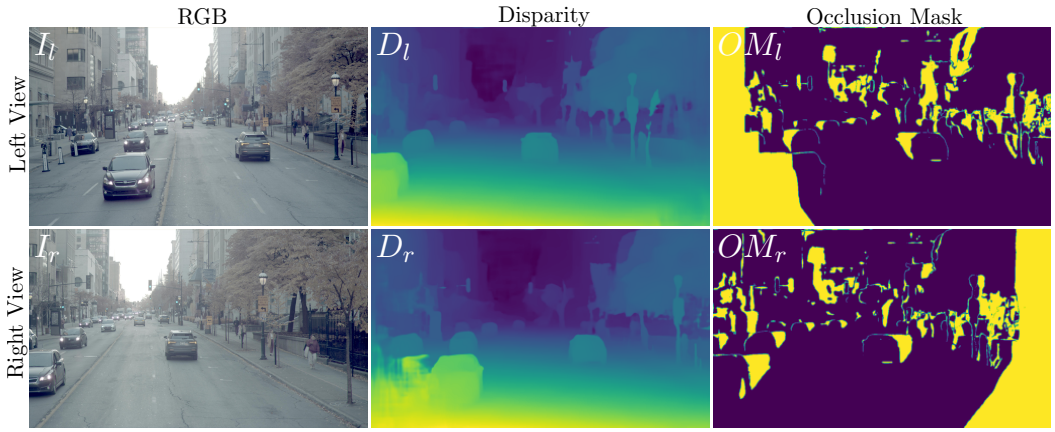


Fig. 7.5: Proposed occlusion masks for the stereo data. We generate OM_l and OM_r based on the consistency assumption that for non-occluded regions, the result of warping the left disparity D_l to the right view should match D_r and vice versa.

Occlusion Masks. Changes in camera view (from left to right stereo camera) and dynamic objects can result in occlusions. Although tracking methods should be robust to changes in appearance between frames, extreme occlusions can be detrimental to the training process. Drastic appearance shifts and occlusion can occur for large baselines, as shown in Figure 7.5. To handle this issue, we use an occlusion mask to discard objects that become heavily occluded.

Specifically, we assume that for non-occluded regions, the disparity in one view should be consistent with the disparity in the other view. We first compute the disparity maps, D_l and D_r, for the left and right views, respectively. Next, we warp D_l to the right view, obtaining $\hat{D}_r$. Subsequently, if the disparity difference for a pixel is above τ_{occ}, that pixel is marked as occluded, that is,

$$\hat{D}_r = \mathcal{W}(D_l, D_r) \tag{7.6}$$

$$\Delta_r = |\hat{D}_r - D_r|,\ OM_r = \begin{cases} 1 & \Delta_r \geq \tau_{occ} \\ 0 & \text{otherwise} \end{cases}, \tag{7.7}$$

where the function $\mathcal{W}(D_l, D_r)$ bi-linearly warps D_l to the right view using disparity D_r and OM_r denotes the occlusion map in the right view. Similarly, we compute the occlusion mask for the left view (OM_l) and discard objects with more than 50% occluded pixels in either OM_r or OM_l from the training data.

Discriminative Feature Extraction

With our differentiable assignment in hand, we train a feature extractor tailored to multi-object tracking in an end-to-end fashion. We find that extracting features from the detectors

fails for object instance association as *object detector features are instance-agnostic and not sufficiently discriminative*, as illustrated in Figure 7.2. To this end, we slightly modify existing object detection architectures [Ren+15] for our purpose; see Figure 7.3. As input, we feed the RGB image and the detection boxes from the separate detectors to the model. The RGB image is initially processed as a whole through a feature pyramid network [Lin+17] with ResNet50 [He+16] backbone; then, the RoI pooling layer extracts the context-aware object features using the detection bounding boxes. Note that this contrasts with directly cropping the object region and then extracting the features [BHM21], resulting in the complete loss of informative contextual information. In the next step, we further process the extracted features with an RoI enhancer module consisting of a stack of convolution and non-linearity layers to obtain an instance-specific object representation specialized for the association task. Finally, the enhanced features are projected to an embedding space $\in R^{256}$ using a small MLP network. The resulting embeddings x_i are used to construct the cost matrix in Equation (7.3).

Training Loss. Assuming we have the correspondences between the detections in the reference and the target frame (from pseudo-labels), we train our model using negative log-likelihood $\mathcal{L}_{NLL}$ and triplet loss $\mathcal{L}_{trip}$, where

$$\mathcal{L}_{NLL} = -\sum_{(i,j)\in\mathcal{A}} \log(P_{i,j}), \tag{7.8}$$

with $\mathcal{A}$ being the set of association pseudo-labels between detections in I_r and I_t.

The additional triplet loss $\mathcal{L}_{trip}$ helps in learning more discriminative features. Specifically, this loss minimizes the distance between the anchor and the positive samples and maximizes the distance between the anchor and the negative samples up to a margin of m, that is

$$\mathcal{L}_{trip}(a,p,n) = max\{d(a_i,p_i) - d(a_i,n_i) + m, 0\}, \tag{7.9}$$

where a, p, and n stand for anchor, positive and negative samples, and $d(x_i, y_j) = |x_i - y_j|$. For this purpose, we select the anchor from I_r and choose the positive and the negative samples from I_t. The final loss is the weighted sum of the terms above, that is

$$\mathcal{L}_{train} = \alpha\, \mathcal{L}_{trip} + \beta\, \mathcal{L}_{NLL}, \tag{7.10}$$

where α and β are training hyperparameters.

Implementation Details

We employ a feature pyramid network [Lin+17] based on ResNet50 [He+16] as the backbone with 256 feature channels and initialize it with pre-trained weights on the COCO dataset [Lin+14]. The RoI pooling layer resizes the extracted regions to a fixed resolution of 21×21. The RoI enhancer module consists of a 4-layer convolutional network with Group Normalization and ReLU non-linearity. The output of this block is flattened and projected to the final embedding $x \in R^{256}$ using a two-layer MLP network and ℓ_2 normalization.

We train our model using SGD optimizer with an initial learning rate of 2×10^{-4}, a momentum of 0.9, a weight decay of 10^{-4}, and a batch size of 8 on a single A100 GPU.

We pre-train our model on temporal and multi-view driving data described in subsection 7.3.2 where we resize the images to the same width as KITTI [GLU12], keeping the aspect ratio unchanged. Additionally, we fine-tune our model on the training set of the datasets used for evaluation. We did not find additional data augmentation beneficial to the final performance.

As training hyperparameters, we set the α and β in Equation (7.10) to 1.0 and 0.5, respectively. During inference, we define the cost matrix as the combination of appearance similarity and IoU, that is

$$C_{inf} = \sigma C_{sim} + (1 - \sigma) C_{\mathrm{IoU}}. \tag{7.11}$$

The IoU information serves as a location prior that helps the model in cases where there are similar objects present in the scene. The relevance of IoU information highly depends on the data frame rate. Therefore, we use $\sigma = 0.7$ and $\sigma = 1$ for the test data captured at higher (e.g., 10 FPS) and lower (e.g., 2 FPS) frame rates, respectively.

7.3.2 Experimental Setup

In this section, we describe the datasets, evaluation metrics, and assess S^{3}Track and relevant baselines on four autonomous driving datasets. We confirm our architecture choices with ablation experiments.

Wide-baseline Stereo Pre-training Data. We capture a training dataset with four 8MP HDR sensors placed at 3m height on a test vehicle with baseline distance (from reference camera ($cam0$) as 0.7m, 1.3m, and 2m for $cam01$, $cam02$, and $cam03$, respectively. The primary configuration during data capture uses four front-facing 8MP 20-bit HDR sensors (AR0820) with 30 degrees horizontal field of view lenses mounted at the height of approximately 3m from the ground and distributed over a 2m baseline. During snow and rain captures, the

Fig. 7.6: Temporal cues used during the pre-training of our method. On our pre-training data, we use optical flow to align the detections in the reference (I_r) and target (I_t) frames. The association pseudo-labels are visualized by the color of bounding boxes in the first two rows.

sensors are mounted behind the windshield at a height of around 1.5m. In all cases, the cameras are mounted using a custom-made mounting plate to ensure that the cameras are attached rigidly and that there is no significant orientation difference between each pair. Calibration for the multi-baseline stereo was performed in two phases: lab-based offline intrinsic parameter estimation and on-site calibration using charts with clearly detectable patterns. Calibration captures were done while the vehicle was static and either in neutral or with the engine turned off to reduce any artifacts due to camera vibration and rolling shutter. Data capture was performed over multiple days to collect sufficient variety in weather, illumination, and scene. A total of 52 hours of data were collected with the capture scenes, including downtown and highways, under varying illumination conditions, including noon with the sun directly above, dusk with the sun near the horizon (direct light on the sensor), and night. Moreover, data were collected covering clear, rainy, and snowy weather conditions. We will release a subset of 2 hours of driving data, evenly distributed for different conditions, including data captured during day, night, dusk, day+rain, day+snow, night+rain, and night + snow, in both downtown and highway traffic conditions. To obtain detections on this dataset, we use a FasterRCNN meta-architecture [Ren+15] with ResNet50 as the backbone and train it on the annotated driving dataset. The visual samples from our pre-training data are shown in Figures 7.6 and 7.7.

KITTI 2D tracking dataset [GLU12] consists of 21 training and 29 test videos collected at 10 FPS deploying sensors mounted on top of a driving car. We fine-tune our model on the train set and evaluate it on the test set using the detections obtained from PermaTrack [Tok+21].

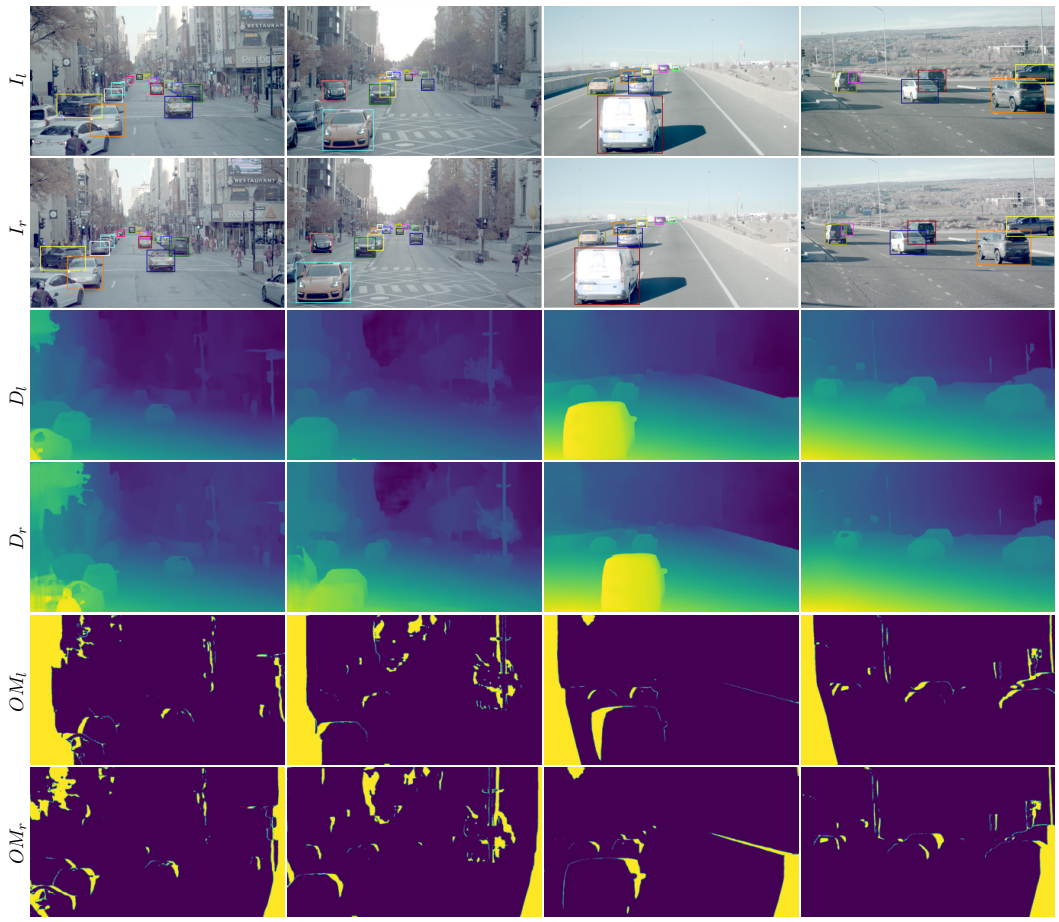


Fig. 7.7: Multi-view cues ($cam0\&1$) used during pre-training. When using stereo data, we use disparity (D_l) to align the bounding boxes from the right image (I_r) to the left view (I_l). Additionally, occlusion masks OM_l and OM_r are utilized to discard objects that are less than 50% visible in one of the views. In the first two rows, we see the left and right RGB images with the association pseudo-labels visualized with bounding box color.

Waymo dataset [Sun+20a] is a large-scale corpus consisting of 798 training and 202 validation sequences each with a duration of 20 seconds at 10 FPS. We use the data captured by the front camera for fine-tuning and evaluation.

nuScenes [Cae+20] includes 700 training videos which are annotated for 3D MOT at 2 FPS. Due to lower annotation frequency, this dataset has a larger appearance change compared to the KITTI and Waymo datasets. We extract the 2D tracking labels from the 3D annotations using the scripts provided by the dataset authors and use 70 percent of the data for fine-tuning and 30 percent for validation.

Argoverse dataset [Wil+21] also provides data for 3D tracking with a training set of 65 and a validation set of 24 videos. Using the script provided by this dataset, we extracted the 2D

	Method	**HOTA**	DetA	**AssA**	DetRe	DetPr	**AssRe**	**AssPr**	LocA
Unsup.	SORT [Bew+16]	71.2	71.6	71.8	74.8	83.5	74.4	88.2	84.8
	IOU [BES17]	74.0	77.4	71.5	81.2	85.8	74.0	88.5	86.9
	UNS20regress[BHM21]	62.5	61.1	65.3	67.7	73.8	69.1	83.1	80.3
	OC_SORT [Cao+22]	76.5	77.3	76.4	80.6	86.4	**80.3**	87.2	87.0
	S^3Track (ours)	**76.6**	77.5	**76.5**	81.3	85.9	79.6	**88.4**	86.9
Supervised	FAMNet [CL19]	52.6	61.0	45.5	64.4	78.7	48.7	77.4	81.5
	CenterTrack [ZKK20]	73.0	75.6	71.2	80.1	84.6	73.8	89.0	86.5
	mmMOT [Zha+19]	62.1	72.3	54.0	76.2	84.9	59.0	82.4	86.6
	LGM [Wan+21]	73.1	74.6	72.3	80.5	82.1	76.4	84.7	85.9
	EagerMOT [KOL21]	74.4	75.3	74.2	78.8	86.4	76.2	91.1	87.2
	DEFT [Cha+21]	74.2	75.3	73.8	80.0	84.0	78.3	85.2	86.1
	PermaTrack [Tok+21]	78.0	78.3	78.4	81.7	86.5	81.1	89.5	87.1
	RAM [Tok+22]	**79.5**	78.8	**80.9**	82.5	86.3	**84.2**	**88.7**	87.1

Tab. 7.1: Tracking Evaluation on the KITTI test set [GLU12] for the *Car* category. In **bold**, we only show the metrics relevant for measuring the association performance. All metrics are in percentage. The proposed S^3Track achieves the best performance in most of the metrics in the unsupervised (Unsup.) category and fares better than most recent approaches in the supervised category.

tracking labels at 5 FPS. We fine-tune on the training data and report the performance on the validation set.

Evaluation Metrics. While a large set of metrics has been proposed to evaluate MOT [Lui+21; Ris+16; BS08], some existing metrics, including MOTA, are biased towards detection accuracy, hence not indicative in the context of evaluating association which is the focus of our work. The most relevant metrics for measuring association performance are the Association Accuracy (AssA), Association Precision (AssPr), Association Recall (AssRe) [Lui+21], and the IDF1 score [Ris+16]. For completeness, we report the conventional metrics from the KITTI tracking benchmark, including Detection Accuracy, Precision, Recall (DetA, DetPr, DetRe), and the HOTA score, which combines the detection and association performance into a single number [Lui+21].

Experimental Results

We evaluate our method on the four autonomous driving benchmarks discussed above. On KITTI [GLU12], we compare our approach to existing supervised baselines and four unsupervised methods. Like our work, the unsupervised methods do not use video-level association annotations and assume the availability of detection bounding boxes, while the *supervised methods are trained using track labels*. [Bew+16; Cao+22] utilize variants of Kalman filtering for modeling the object motion, while the tracker in [BES17] works purely based on IoU information. In [BHM21], the authors use a motion model based

Fig. 7.8: Qualitative Tracking on KITTI [GLU12]. We compare unsupervised S^{3}Track and supervised PermaTrack [Tok+21] on unseen sequences. The track IDs are visualized with color coding and the unique number inside each bounding box. Our method shows robust performance under heavy occlusion (see zoom-ins on the occluded regions). In both scenes, S^{3}Track correctly handles the heavy occlusion maintaining the track IDs, while PermaTrack[Tok+21] suffers from several ID switches and fragmentation.

on bounding box information and an appearance model, and the self-supervised objective aims to enforce consistency between the motion and the appearance model outputs. These methods require small object motion to work well, an assumption often violated in driving scenarios, especially with a low capture frame rate.

Table 7.1 reports tracking evaluations for the 'Car' class on the KITTI [GLU12] test set. Together with OC_SORT [Cao+22], our method outperforms other unsupervised baselines and even multiple supervised methods such as CenterTrack [ZKK20], EagerMOT [KOL21], and DEFT [Cha+21] – which have access to track labels – and achieves comparable results with PermaTrack [Tok+21] *without using any video-level association labels*. We highlight that [Cao+22] is a purely motion-based approach and can be complementary to our proposed appearance-based method. In Figure 7.8, qualitative examples show that S^{3}Track outperforms PermaTrack in complex occlusion scenarios.

In Table 7.2, we discuss MOT evaluations for the 'Car' category on several recent automotive datasets, namely Waymo [Sun+20a], nuScenes [Cae+20], and Argoverse [Wil+21]. The results for other baselines in Table 7.2 are obtained using the published code from the respective authors. The evaluations validate that our S^{3}Track performs well across all

Method	Waymo [Sun+20a]				nuScenes [Cae+20]				Argoverse [Wil+21]			
	AssA	IDF1	AssRe	AssPr	AssA	IDF1	AssRe	AssPr	AssA	IDF1	AssRe	AssPr
SORT [Bew+16]	62.2	71.9	63.9	93.1	56.5	66.1	59.2	82.1	63.2	75.5	63.9	93.1
IOU [BES17]	72.1	79.4	73.2	94.5	60.8	71.5	69.3	72.6	70.1	80.2	73.2	94.5
OC_SORT [Cao+22]	72.4	79.4	74.1	93.5	65.6	72.3	71.2	81.5	74.1	82.8	74.1	93.5
S^3Track (ours)	**77.8**	**83.7**	**78.5**	**97.7**	**73.4**	**81.9**	**79.0**	**87.7**	**77.8**	**83.7**	**78.5**	**93.5**

Tab. 7.2: Evaluation on Waymo [Sun+20a], nuScenes [Cae+20], and Argoverse [Wil+21] datasets for the *Car* category. Our method consistently outperforms the other unsupervised methods with a considerable margin of about 4-8 points on AssA across all datasets. As can be seen from the results, our method shows a robust performance when processing low frame rate videos as in nuScenes dataset. This is contrary to the motion-based models [Bew+16; BES17; Cao+22], where the association accuracy decreases at lower frame rates with an increased object motion.

datasets and scales well to larger datasets with varying data characteristics such as weather conditions and frame rate. This is in contrast with motion-based models [Cao+22; Bew+16; BES17], where the performance considerably drops at lower frame rates.

Ablation Studies

We conduct ablation experiments that validate the effectiveness of different components of our method. For all experiments, we train on the proposed wide-baseline stereo driving data and evaluate on the (now unseen) KITTI training set where the detections are available.

Table 7.3 shows the contribution of the main components of S^3Track. In the first row, we assess the importance of the RoI pooling mechanism for extracting the context-aware object features by first extracting object patches and then extracting features using a ResNet50 network. Next, we evaluate the effect of the RoI enhancer module. In this experiment, we directly perform average pooling on the extracted RoI features to obtain the final embeddings ($x_i \in R^{256}$). In the third ablation experiment, we inspect the role of the soft assignment, which enables the end-to-end training of our model. Here, we compute the embeddings similar to the previous experiment without further training and use pre-trained object detection weights on the COCO dataset [Lin+14].

Method	AssA	AssRe	AssPr	IDF1
S^3Track	**96.1**	**97.5**	**97.7**	**97.6**
w/o RoI Pooling	92.0	94.2	95.8	94.6
w/o RoI Enhancer	92.9	95.1	96.2	95.3
w/o soft assignment	83.5	88.2	90.1	89.1

Tab. 7.3: Ablation Experiments. We confirm the effectiveness of the RoI-based feature extraction, RoI enhancing module, and soft object assignment with optimal transport. To quantify the relevance of each component, we run an experiment without each component and report the change in the association performance.

Impact of the Distance Function. For the experiments presented in Tables 7.1 and 7.2, we use cosine distance as the measure of closeness between object embeddings. In Table 7.4, we provide experimental results when training the model with an alternative ℓ_2 distance and using a matching network for predicting the similarity score (instead of using a pre-defined similarity/distance function). The architecture of the matching network is an MLP consisting of 3 linear layers with 1024, 256, and 1 output channels, respectively (the output of the last layer is the similarity score). We use ReLU non-linearity between linear layers. The input to the matching network is the concatenation of different object-pair embeddings; this network is expected to learn the function measuring the embedding similarity. We observe that using the learnable function in the matching network under-performs the ℓ_2 and cosine distance functions.

Distance Function	AssA	AssRe	AssPr	IDF1
Cosine	**94.8**	**96.5**	**96.8**	**96.9**
ℓ_2	94.4	96.2	96.6	96.7
MLP	90.7	92.4	95.7	93.9

Tab. 7.4: Ablation experiments evaluating the choice of the distance function. An embedding distance using ℓ_2 and cosine distance performs better than using a matching network (MLP) for learning the object similarity score. The experiments are conducted using temporal data at 5 FPS.

Influence of Pre-training Cues. We investigate the influence of different training cues on the proposed method using the *unseen* KITTI [GLU12] training set (with detections available).

In Table 7.5, we study the impact of frame rate when using temporal pre-training cues. We observe that a high frame rate achieves sub-optimal performance as there is not enough change in object appearance. A very low frame rate also decreases the accuracy due to extreme appearance changes which differ from the testing data. These findings also transfer to the stereo configuration.

FPS	AssA	AssRe	AssPr	IDF1
15	83.2	85.3	94.9	88.5
5	**94.8**	**96.5**	**96.8**	**96.9**
1	93.4	94.8	97.3	95.7

Tab. 7.5: Ablation experiments that investigate the impact of frame rate for temporal pre-training data.

Table 7.6 assesses the method when training with different data types, including temporal video data, stereo data, and a combination of both. To evaluate the influence of stereo cues, we tested with data from the three different camera pairs with varying baseline sizes, similar to the ablation experiment on distance functions. Training with data from $cam0\&1$ yields better accuracy. The larger baseline in $cam0\&2$ and $cam0\&3$ results in a higher object appearance shift between the two views, which, in this case, is detrimental to the accuracy due to the domain gap with the KITTI data used for evaluation. Moreover, we find that the combination of temporal and stereo data is beneficial for training a better appearance model.

Data	AssA	AssRe	AssPr	IDF1
$cam0\&1$	95.1	96.7	97.1	97.1
$cam0\&2$	94.9	96.1	97.6	96.7
$cam0\&3$	93.3	94.9	96.5	95.5
temporal	94.8	96.5	96.8	96.9
temporal + $cam0\&1$	**96.1**	**97.5**	**97.7**	**97.6**

Tab. 7.6: Ablation experiments for temporal and stereo pre-training cues. Here, the temporal pre-training data is sampled at 5 FPS.

In Table 7.7, we study the effect of the pre-training step on the final association performance on the KITTI [GLU12] test set. In S^3Track^+, we first pre-train the model on our driving dataset and then fine-tune it on the KITTI training set. In S^3Track^-, we initialize the model with pre-trained weights on the COCO dataset [Lin+14] and directly train the model on the KITTI training set.

Method	Pre-training	HOTA	AssA	AssRe	AssPr
S^3Track^+	Yes	**76.6**	**76.5**	**79.6**	**88.4**
S^3Track^-	No	75.2	73.9	76.3	88.0

Tab. 7.7: Ablation on the impact of pre-training on our driving dataset, evaluating on KITTI [GLU12] test set.

7.3.3 Summary

We propose S^3Track – a self-supervised method for multiple object tracking that operates without any video-level track labels aiming at alleviating the expensive process of data annotation for MOT. With object bounding boxes from an accurate object detector in hand, our model performs MOT by learning the object associations over the video frames. To

this end, we propose a soft differentiable assignment approach, which allows us to train our model end-to-end using the association pseudo-labels acquired from motion information in temporal and multi-view video data. The differentiable assignment makes it possible to learn context-aware object features that are specialized for the association step. We validate our method on four autonomous driving benchmarks and demonstrate favorable performance across different datasets achieving on-par or better performance than other unsupervised methods. Future directions include jointly learning association and motion trajectory and exploring memory-based approaches for merging object appearance over multiple frames.

7.4 Self-supervised Learning for Test Time Adaptation on Video Data

Most modern computer vision applications follow the general two-step paradigm of first training a model on a large dataset and then deploying it on unseen test data. However, the majority of these applications are still designed under the assumption that training and test data have been sampled from the same distribution. As this assumption is frequently violated in the real world, the applicability of these models can often be very limited [HD19b; Rec+19]. It is thus important to seek strategies for adapting pre-trained models to the test data. However, *supervised* fine-tuning on the domain from which the test data has been sampled is often unfeasible. Even if the distribution from which the test data has been sampled is accessible, labeling can be cumbersome and expensive. This issue is particularly relevant for video tasks, which often require per-frame pixel-wise labels (e.g., [Dav+20; Pon+17]). Nonetheless, videos contain a wealth of information, especially if we assume that some (unlabeled) frames from the test distribution are available before actually performing the evaluation. Consider the practical case in which a drone for aerial photography is deployed in an unseen environment; snowy weather, for instance. It is then reasonable to assume that the first few seconds of its unlabeled video feed can be used to adapt its models to the surroundings in which it will soon be operating. Intuitively, collecting unlabeled sample videos from the new domain is a significantly simpler task than obtaining labeled data.

Motivated by these observations, we explore how unlabelled video data can be exploited in order to adapt pre-trained models to the distribution to which the test set belongs. Self-supervised methods are of particular interest for our scenario, as their objective allows for "fine-tuning without labels." In our evaluation, we address the task of video object segmentation, also known as dense tracking [Cae+17], as it has often been used to compare self-supervised methods trained on video data [Von+18; LLX20; JOE20]. In this task, the pixel-wise mask of the target object is provided at test time in the first frame of the video, and the goal is to track the object of interest throughout the video sequence by providing per-frame masks.

Besides not having access to labeled data from the test distribution, in our experiments, we consider another important condition: We assume to have received an already-trained model and not have access to either its training routine or to the data it has been trained with. This scenario has recently become of great interest because of the increasingly prohibitive cost of training large-scale state-of-the-art models [Bom+21].

Recently, several works in the image domain have studied a (*de facto*) similar setup under the name of *test-time adaptation*. [Nad+20; Sch+20; Sun+20c; Wan+20a]. However, these methods often rely on batch normalization and implicitly assume the availability of batches with elements sampled i.i.d. from the test distribution to be used for adaptation. In contrast, this assumption is inevitably violated when adapting the model with data originating from a video stream, like in our case. In this work, we re-purpose several test-time adaptation methods used in the image domain and experiment to which extent they can be useful with video data. In particular, we are interested in evaluating how well we can exploit unlabeled videos by using self-supervised objectives to improve the test performance on the downstream task. To this end, we investigate two distinct scenarios of arbitrary and severe domain shifts. In the first case, we perform the test-time adaptations on unseen videos, whose originating distribution may differ from the distribution of the training data in arbitrary and unknown ways. In the second case, we impose severe (but controlled) domain shifts by artificially adding perturbations to the video frames.

In our experimental setup, we consider two outlines for the offline and online applications. In offline applications such as video editing, all the video frames are available beforehand and can be exploited for adaptation. However, this is clearly not possible in online applications such as autonomous driving, as real-time inference is required. Nevertheless, it is still reasonable to assume that we have access to a limited amount of unlabeled data from the target domain. Therefore, we utilize all the video frames for test-time adaptation in the first scenario and only use a fraction of the frames in the latter.

In summary, our contributions are two-fold: First, we introduce a novel problem formulation to investigate the potential of using unlabeled video data for test-time adaptation in a self-supervised manner. Then, we perform an extensive evaluation to understand the behavior of current state-of-the-art dense tracking methods in the presence of several types of domain shifts and the impact of test-time adaptation in alleviating their detrimental effects.

7.4.1 Problem Formulation and Methods

This section discusses our proposed problem formulation, followed by an overview of the utilized baselines and test-time adaptation algorithms. Our primary focus lies on studying the impact of covariate shifts in the task of self-supervised dense tracking and the possible remedies utilizing unlabeled video data. We are interested in studying ways to adapt a pre-trained model to the target data distribution without altering the training regime. This setup is beneficial due to the many practical use cases in real-world conditions. Inspired by test-time adaptation literature from the image domain [Sun+20c; Wan+20a; Nad+20;

Sch+20], we explore utilizing unlabelled video data for addressing the problem of covariate shift in the video domain.

Test-time adaptation methods in the image domain usually assume the availability of a *diverse* batch of unlabeled data from the target distribution during inference. These data are used for further fine-tuning the model with an unsupervised or self-supervised objective. As a video contains much more information than a single image, in this work, we study the extent to which unlabeled video frames can be utilized for test-time adaptation. We note that the definition of *domain* in the literature is relatively imprecise. For example, it is unclear if we consider a dataset as a single domain or a combination of multiple domains (each class forming a cluster can be viewed as a separate domain). Hence, we initially contemplate a hypothesis where each video can be considered as an individual domain. Next, we enforce domain shift by manually adding various perturbations to the test videos [HD19b]. To this end, we ask the following questions:

- Assuming each video represents a specific domain, how effective are the current test-time adaptation methods when applied to the task of dense tracking in videos?

- Considering the self-supervised setups for dense tracking, can further fine-tuning the model on the target video (essentially overfitting to a specific video domain using the self-supervised objective) improve the performance on the downstream task?

- In the case of clear domain shift such as noisy data, how effective are these adaptation methods for recovering the performance in self-supervised dense tracking tasks?

To answer these questions, we experiment with modified variants of three recent approaches for test-time adaptation from the image domain, namely **Prediction-time BN** [Sch+20; Nad+20], **TENT** [Wan+20a], and **TTT** [Sun+20c]. Prediction-time BN introduces the idea of updating the batch normalization (BN) layer statistics using the test data to tackle the impact of covariate shift on model performance. As mentioned in Section 4.2.7, the BN layer relies on accumulated statistics from the training data during inference, which can lead to performance degradation when there is a distribution shift between the training and test data. To address this, TENT and TTT propose more intricate setups involving the updating of both the BN statistics and network weights.

As our self-supervised dense tracking baselines, we chose two state-of-the-art methods of VideoWalk [JOE20] and MAST [LLX20]. In the following, we explain the utilized test-time adaptation methods as well as the selected dense tracking baselines.

Test-time Adaptation

Prediction-time BN [Nad+20; Sch+20] suggests replacing the statics of the normalization layer (μ, σ) with the ones estimated from the test data. [Nad+20; Sch+20] observe that in a scenario where there is a domain shift between the training and testing data, it is sub-optimal to normalize the activations with the μ and σ estimated from the training data. Assuming a batch of data from the target distribution is available at inference time, they propose to either replace [Nad+20] or update [Sch+20] the normalization statics with the ones computed from the test data.

TENT [Wan+20a] algorithm proposes to update the normalization statistics as well as the shift and scale parameters γ and β in the BN layer and adapt the feature modulation to the target data distribution. In [Wan+20a], the authors use entropy minimization as their optimization objective:

$$H(\hat{y}) = -\sum_{c} p(\hat{y}_c)\ log\ p(\hat{y}_c) \tag{7.12}$$

where $p(\hat{y}_c)$ is the network output probability for class c. As mentioned earlier, in our adopted variant of this method referred to as TENT*, we experiment with the self-supervised objectives (Equation 7.17 and Equation 7.20) instead of the entropy loss in Equation 7.12.

Test-time Training (TTT) [Sun+20c] alters both the training and inference procedures. In [Sun+20c], the authors modify the architecture to include a shared backbone as well as two separate heads for the main task (image classification) and a self-supervision objective, namely rotation classification. The model is then trained with the standard image classification objective together with the auxiliary loss in a multi-task setup. During the test phase, the model is further fine-tuned using the auxiliary objective. This way, the parameters of the shared backbone are modified and adapted for the target distribution, but the classification head remains unchanged. Therefore, the auxiliary head is utilized to adapt the backbone to the target data distribution and mitigate the impact of the covariate shift between train and test data distributions.

However, our setup is different from these methods, as discussed in the following: First, the aforementioned methods are developed for image classification and assume that a diverse batch of data from the target distribution will be available at test time. In our setup, each video is considered an individual domain, and the frames sampled from a single video comprise the batch, meaning the batch might not contain enough diversity. Unlike the prediction-time BN scenario in [Nad+20; Sch+20], the captured statistics from a video sequence may not be diverse enough, so replacing the normalization statistics in the

normalization layer with those collected from the video frames might hurt the performance. Therefore, we experiment with different momentum values:

$$\hat{x} = (1 - \alpha) \times x_{old} + \alpha \times x_{new} \tag{7.13}$$

where x_{old} is the statistics estimated from the training data, x_{new} is the statistics computed from the video at hand, and $\alpha \in [0, 1]$ is the momentum. Second, these methods build on top of models trained in a supervised manner while we examine baselines that are trained in a self-supervised fashion. Third, we use a modified version of TENT [Wan+20a] where the self-supervised objective substitutes the entropy loss. TENT minimizes the entropy of the class prediction, while in dense tracking, the first mask is required for computing the pixel-wise label probabilities. As we aim to solve fully unlabeled test-time adaptation, we utilize the self-supervised objective instead of entropy minimization. We refer to this adapted version as TENT*.

Self-supervised Dense Tracking

Recently, self-supervised methods for dense tracking have significantly improved, achieving impressive performance comparable to supervised counterparts [LLX20; JOE20]. One of the earliest works in this area was [Von+18], where the authors proposed to learn the correspondences based on video colorization. This algorithm has been the basis for many other approaches, such as [LX19b; LLX20]. In this method, a self-supervised objective for correspondence matching is defined based on colorizing the frames in a video. To this end, consider a colored reference frame where each pixel has a value $c_i \in R^d$ (colors are quantized to d bins) and a grayscale target image. The colors in the target frame are quantified as:

$$y_j = \sum_i A_{ij} c_i \tag{7.14}$$

where A is a similarity matrix computed as:

$$A_{ij} = \frac{\exp(f_i^T f_j)}{\sum_k \exp(f_k^T f_j)} \tag{7.15}$$

In Equation 7.15, f represents the image features computed by a neural network which is trained by minimizing the following objective.

$$\text{Loss} = \sum_j \text{Cross\_Entopy}(y_j, c_j) \tag{7.16}$$

Here, c_j is the correct quantized color, and y_j is the predicted color class by the model, based on Equation 7.14. During inference time, the learned features are utilized for various applications such as pose tracking and video object segmentation.

MAST [LLX20] is one of the state-of-the-art methods based on colorization. In [LLX20], the authors make several improvements to [Von+18] as explained in the following. First, they suggest using lab color space instead of RGB due to less correlation between color channels. Second, they enhance the architecture by incorporating a memory bank and employing an attention mechanism to retrieve the color in each target frame from multiple past frames using the attention weights. Third, they propose to replace the classification objective in Equation 7.16 with regressions as:

$$\text{Loss} = \frac{1}{n}\sum_i \begin{cases} 0.5(\hat{I}_t^i - I_t^i)^2 & \text{if } |\hat{I}_t^i - I_t^i| < 1 \\ |\hat{I}_t^i - I_t^i| - 0.5 & \text{otherwise} \end{cases} \tag{7.17}$$

where n is the number of pixels and $\hat{I}_t^i$ and I_t^i are the estimated and the actual color values for the i_{th} pixel, respectively. The intuition is that quantizing the colors to a limited number of classes causes loss of important information and leads to sub-optimal performance, whereas using regression preserves all the color information.

VideoWalk [JOE20], in contrast to MAST, develops a framework for correspondence matching based on learning the patch-wise similarities across the video frames. In this algorithm, a space-time graph is formed by dividing each frame into multiple nodes (patches) and computing the edge weights based on a similarity metric between the neighboring nodes (across time and spatial dimensions). Consequently, the task of finding the correspondences across the video frames is devised as a contrastive random walk with patch-wise affinities providing the transition probabilities.

Assume q_t^i is the feature embedding of i_{th} node/patch at time step t. The Affinity matrix between every two nodes in consecutive frames can be calculated as follows:

$$A_t^{t+1}(i,j) = \frac{\exp(\langle q_t^i, q_{t+1}^j \rangle / \tau)}{\sum_{l=1}^N \exp(\langle q_t^i, q_{t+1}^l \rangle / \tau)} \tag{7.18}$$

where τ is a temperature parameter. Subsequently, the long-range affinities between the nodes from non-consecutive frames are computed as:

$$\hat{A}_t^{t+k} = \prod_{i=0}^{k-1} A_{t+i}^{t+i+1} = P(X_{t+k}|X_t) \tag{7.19}$$

The goal is to train the embeddings such that higher weights are assigned to the edge between similar patches so that the random walk likely follows the path of the corresponding nodes.

To achieve this goal, [JOE20] uses an objective based on cycle consistency and creates a palindrome of video frames so that for each node at the first time step, we know the target node at the end of the walk. This objective can be formulated as follows:

$$\text{Loss} = \text{Cross\_Entropy}(\hat{A}_t^{t+k}, Y_t^{t+k}) \tag{7.20}$$

where Y_t^{t+k} is the actual corresponding node which is known as a result of the palindrome setup and the cycle consistency.

7.4.2 Experimental Setup

In this work, we experiment with DAVIS2017 [Pon+17] and TAO-VOS [Voi+20] datasets, two standard benchmarks for evaluating dense tracking methods.

DAVIS2017 [Pon+17] validation set consists of 30 videos with an average duration of 3.4 seconds. As the videos in DAVIS are somewhat short, especially for the online setup where half of the video frames are used for adaptation, we further benchmark this setup on a sub-set of **TAO-VOS** [Voi+21; Dav+20]. The videos in this dataset are considerably longer than DAVIS2017, with an average length of 36.7 seconds. Therefore even when using half of the frames for the evaluation, we end up with longer videos than DAVIS2017. We selected this subset based on two criteria: each video contains at least 1000 frames and at most 2 target objects. The second condition is a practical consideration, as the current self-supervised methods do not work well in scenes with many objects. Therefore, we resort to relatively more straightforward videos with more frames, allowing us to use a subset of data for test-time adaptations.

For the adaptation methods that require tuning the model parameters, we adhere to the training setup outlined in the original paper of each deployed self-supervised algorithm. Specifically, we utilize the identical optimizer with the learning rate maintained at the last stage of the training process. We continue training the model until it reaches convergence, typically achieved within approximately 200 iterations.

We report the standard evaluation metrics of dense tracking task, *Region Similarity* and *Contour Accuracy* ($J\&F$) scores [Pon+17]. J refers to the intersection-over-union between the model prediction and the ground-truth and F measures the quality of the estimated object boundaries.

Fig. 7.9: Samples from corrupted data distributions (Gaussian noise and Snow at the top row, Fog and Motion Blur at the bottom row).

Experimental Results

Table 7.8 presents the results for offline and online setups on DAVIS2017 dataset. Each row shows the J and F scores of the baselines in the presence of a specific domain shift, with and without test-time adaptation. In the first block of rows, we investigate the efficacy of test-time adaptation with a self-supervised objective on the test data with arbitrary domain shifts (without any added perturbation). As we are working with self-supervised baselines, a question naturally arises whether further tuning on a specific video is helpful and to what extent it can improve the performance of the downstream task. Next, we study a scenario with a substantial domain shift between the training and testing data distributions. In this respect, we follow the proposed setup in [HD19a] and impose an artificial covariate shift to the video frames. In particular, we experiment with Gaussian noise, Motion Blur, Fog, and Snow perturbations, shown in Figure 7.9. Perturbations are generated according to the level 5 severity as described in [HD19b].

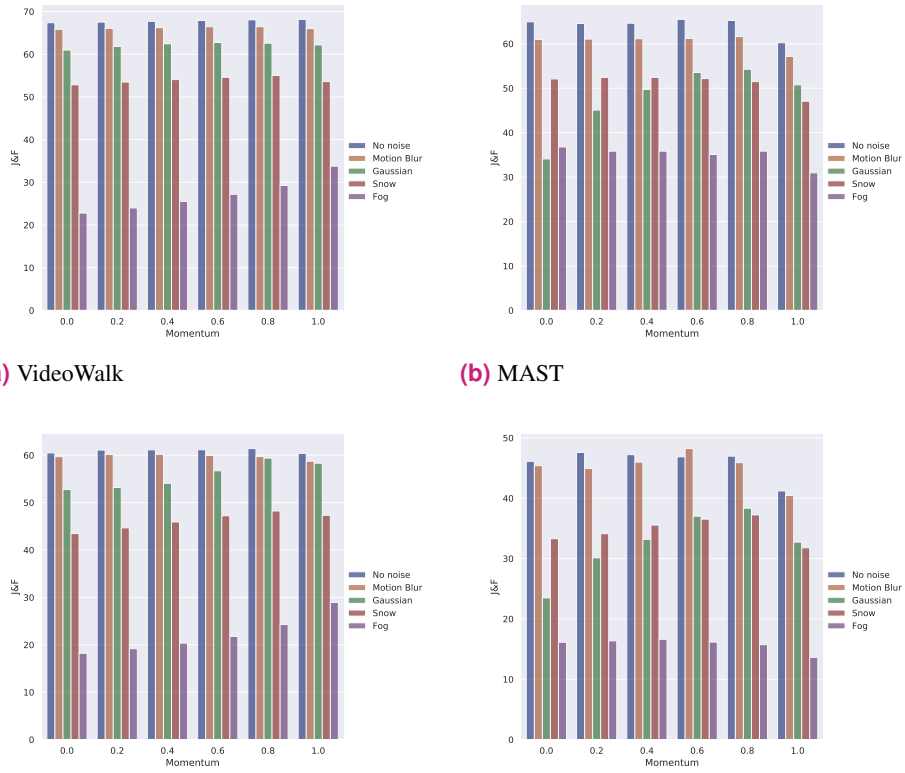
As can be seen from the results in Table 7.8, self-supervised test-time adaptation on the data without perturbation slightly improves the results while considerably decreasing the adverse effect of covariate shift for data with severe perturbations. The behavior in an arbitrary domain shift scenario (without perturbation) implies that in situations with mild distribution shift, overfitting to the current self-supervised objectives does not fully transfer to the downstream task and only marginally improves the performance. However, these methods can successfully adapt the features to the target domain when there is a severe distribution shift between the training and testing data. Interestingly, in most cases, updating the normalization statics (BN column) has an equal or superior positive impact on the

Dense Tracking (Offline)				Dense Tracking (Online)				Test-time Adaptation			
VideoWalk		MAST		VideoWalk		MAST		BN	TENT*	TTT	Noise
J	F	J	F	J	F	J	F				
64.38	*70.40*	*62.95*	*66.94*	*69.46*	*74.43*	*67.11*	*70.85*				—
+1.00	**+0.56**	**+0.47**	+0.62	+0.67	**+0.99**	**+1.04**	**+1.04**	✓			
+1.04	+0.50	+0.32	**+0.65**	**+0.70**	+0.97	+0.20	+0.30		✓		
+1.17	+0.47	+0.09	+0.34	+0.64	+0.84	+0.27	+0.39			✓	
58.40	*63.08*	*32.70*	*35.48*	*64.43*	*67.89*	*41.51*	*43.36*				Gaussian
+1.85	+2.16	**+19.82**	**+20.54**	+2.07	+2.58	**+18.21**	**+19.26**	✓			
+1.91	+2.44	+17.98	+18.77	**+3.73**	**+3.91**	+15.90	+17.17		✓		
+2.67	**+2.97**	+18.06	+18.15	+2.11	+2.20	+15.37	+16.58			✓	
62.97	*68.75*	*58.49*	*63.45*	*67.69*	*72.50*	*64.54*	*69.99*				Motion Blur
+0.69	**+0.51**	**+0.49**	**+0.80**	+1.01	+1.62	**+0.35**	**+0.10**	✓			
+0.41	+0.34	-0.10	+0.13	**+1.04**	**+1.69**	-0.21	-0.22		✓		
+0.18	+0.11	+0.12	-0.18	+0.97	+1.28	-0.58	-0.43			✓	
50.89	*54.77*	*51.12*	*53.08*	*56.44*	*59.20*	*58.51*	*59.68*				Snow
+1.63	+2.78	**+0.83**	**+0.77**	**+2.60**	**+2.80**	+0.51	+0.46	✓			
+1.99	+2.80	+0.14	+0.34	+2.43	+2.52	**+0.77**	**+0.99**		✓		
+2.79	**+3.92**	+0.32	+0.39	+1.98	+1.91	+0.15	+0.38			✓	
19.27	*26.32*	*35.55*	*38.05*	*24.76*	*30.76*	*43.42*	*45.03*				Fog
+11.23	+10.76	0.00	0.00	+11.54	+9.860	0.00	0.00	✓			
+12.01	+12.23	+3.09	+2.66	+9.67	+9.22	+3.83	+3.51		✓		
+18.70	**+18.42**	**+9.85**	**+8.50**	**+14.07**	**+14.21**	**+9.24**	**+9.54**			✓	

Tab. 7.8: J and F scores for VideoWalk [JOE20] and MAST [LLX20] self-supervised dense tracking methods on **DAVIS2017** validation set in **offline** and **online** settings. In the offline mode, all video frames are used for adaptation. In the online setup, we use the first and the second half of the video for adaptation and evaluation, respectively. For each perturbation variant, we compare the accuracy of the baseline model with three test-time adaptation techniques as explained in Section 7.4.1. Results in cursive correspond to absolute metrics, followed by their delta when using one of the test-time adaptation methods. Best results per column are shown in bold.

dense tracking accuracy despite its simplicity. However, we note that Fog perturbation is an exception where both MAST and VideoWalk methods achieve considerably better accuracy with TENT* and TTT algorithms. Furthermore, the results show a similar pattern in offline and online scenarios, suggesting that performing test-time adaptation is beneficial for both circumstances.

For the results shown in column BN, we experimented with different momentum values and updated the normalization statistics according to Equation 7.13. Here the results are provided with the best-found momentum, and additional results can be seen in Figure 7.10. From these plots, we see that partially updating the normalization statistics with those from the target domain alleviates the impact of covariate shift, but completely replacing them (momentum value of 1) can deteriorate the performance. This behavior can be due to a lack of diversity in video frames, resulting in sub-optimal performance when ignoring the information collected from the training data (the old normalization statistics). Moreover, we observe varying trends in the VideoWalk and MAST methods; for example, in Fog perturbation, VideoWalk benefits from updating the normalization statistics, whereas it is better to keep the statistics unchanged for MAST. This can result from different training objectives in these approaches as the self-supervised loss in MAST is purely based on



(a) VideoWalk **(b)** MAST

(c) VideoWalk **(d)** MAST

Fig. 7.10: Ablation on the momentum in prediction-time BN (Equation 7.13) and the impact it has on the performance of VideoWalk and MAST methods under different perturbations. The first and second rows illustrate the results on DAVIS and TAO-VOS datasets, respectively. The diagrams on DAVIS are from the offline setup (we observed a similar trend in the online mode). The results indicate that, except for Fog, it is better to update the normalization statics with a momentum value of less than one in most cases. In VideoWalk, it is better to completely replace the statistics with those collected from the target domain, while in MAST, it is better to keep the statistics unchanged.

color information (Equation 7.17), while VideoWalk additionally utilizes higher-level correspondences between pixel embeddings (Equation 7.20).

As explained in Section 7.4.1, TTT [Sun+20c] and TENT [Wan+20a] approaches fine-tune the network weights. We note that updating the model weights also depends on how the normalization statistics in the BN layer are handled (i.e., training the model when freezing or updating the BN statistics). In these methods, it is assumed that a diverse batch of data is available, but this condition may not hold when sampling the batch from a video sequence. Therefore, we need to consider this factor and carefully treat the BN layer. We experimented with both cases of training with freezing and updating the normalization statistics. The results in Tables 7.8 and 7.9, are with the best-found configuration.

Dense Tracking				Test-time Adaptation			
VideoWalk		MAST		BN	TENT*	TTT	Noise
J	*F*	*J*	*F*				
55.83	*65.17*	*43.68*	*48.59*				—
+0.83	+0.98	**+1.49**	+1.42	✓			
+1.23	+1.52	+1.21	+1.46		✓		
+1.42	**+1.76**	+0.85	**+2.11**			✓	
48.29	*57.25*	*22.34*	*24.64*				Gaussian
+6.51	**+6.78**	**+14.40**	+15.31	✓			
+3.56	+3.90	+13.79	+14.71		✓		
+4.35	+4.56	+14.21	**+15.33**			✓	
55.21	*64.19*	*42.69*	*48.12*				Motion Blur
+0.71	+0.29	**+2.71**	**+2.95**	✓			
+1.01	**+1.09**	+2.32	+1.93		✓		
+0.58	+0.43	+2.65	+2.71			✓	
38.79	*48.16*	*31.53*	*35.07*				Snow
+5.18	+4.35	**+3.94**	+3.95	✓			
+6.48	**+5.88**	+2.82	+2.81		✓		
+6.56	+5.31	+3.87	**+4.11**			✓	
14.47	*21.96*	*14.60*	*17.67*				Fog
+11.11	+10.07	+0.52	+0.49	✓			
+23.64	**+24.62**	+1.63	+2.10		✓		
+22.24	+22.20	**+6.12**	**+5.48**			✓	

Tab. 7.9: J and F scores for VideoWalk [JOE20] and MAST [LLX20] self-supervised methods on a subset of the **TOA-VOS** dataset in an **online setup** when using half of the video for adaptation and evaluation on the second half of the frames. Results in cursive correspond to absolute metrics, followed by their delta when using one of the test-time adaptation methods. Best results per column are shown in bold.

Considering the short duration of DAVIS2017 videos, utilizing half of the video may not provide solid conclusions. Therefore, we also benchmark the baselines on a subset of TAO-VOS, which contains about ten times longer videos than DAVIS2017. We follow the same experimental setup described before. Table 7.9 presents the results for this dataset in the online setting, where we use the first half of the video for adaptation and the rest for evaluation. Furthermore, Figures 7.10c and 7.10d show the performance of the baselines when updating the normalization statistics with varying momentum values, as in Equation 7.13. The results in the BN column in Table 7.9 are obtained using the best-found momentum based on this ablation.

Based on the quantitative results obtained from TAO-VOS, we observe a similar pattern to the findings from DAVIS2017. Test-time adaptation consistently enhances performance in the presence of domain shift. Nevertheless, the extent of improvement varies depending on the specific type of domain shift encountered. These results validate the efficacy of test-time adaptation for short and long videos encouraging further research in this domain.

7.4.3 Summary

In this work, we investigate the role that self-supervision can have in alleviating the harmful effect of distribution mismatch between train and test datasets of video data. We consider two scenarios of practical relevance. One for offline applications, in which the entire video sequence is available in advance. Another is for online applications, in which we are interested in real-time inference and have access to some unlabeled data from the target domain prior to inference. In both cases, we only consider a pre-trained model without having access to the training data. We study the behavior of two recent self-supervised dense tracking algorithms in the presence of several domain shifts, including Gaussian, Motion Blur, Fog, and Snow perturbations. Our experimental results confirm that self-supervised test-time adaptation is an effective method for decreasing the impact of covariate shifts in dense tracking but that the extent of its efficacy largely depends on the specific shifts and algorithms in question. For instance, when dealing with Gaussian noise, updating only the statistics of the batch normalization layer outperforms other more complex approaches. However, for perturbations such as Snow and Fog, updating the network parameters proves to be more advantageous.

8 Conclusion

Despite the excellent performance of deep learning models on academic datasets and evaluations under controlled lab conditions, their performance is not robust enough for unconstrained real-world situations. In this thesis, we investigated the limitations of current deep learning models in computer vision applications across image and video modalities, aiming to better understand these aspects and propose solutions that improve the robustness and applicability of these models in day-to-day scenarios. The following summarizes the key questions and contributions investigated in this work.

Image Domain

cluttered image classifications

In the image domain, we analyzed the performance of learning-based classifiers when handling images with substantial background noise and clutter. We observed a significant decline in classification accuracy under such conditions. To address this issue, we trained an RL agent named SSTN to sequentially transform input images, removing clutter and zooming in on the main content. Our experimental results confirmed this as a highly effective solution for recovering the classifier's accuracy.

curriculum learning

In this context, we demonstrated that the transformations learned by SSTN systematically modify the input data distribution, gradually moving it toward an easier one. As such, the generated data provides a hard-to-easy spectrum that can serve as a curriculum policy. Through experimenting with several strategies for deploying these data at different rates during the training, we showcased how the data generated by SSTN can be leveraged to enhance the training process of the classifier within the framework of curriculum learning.

salient image classification

Based on the intuition developed from these experiments, we considered a more challenging setup where the image clutter can be actual objects. To approach this scenario, we made the assumption that object size correlates with saliency. Accordingly, we designed DQ-SSTN, an RL-based model that learns to zoom in on the largest/primary object in the scene by using object IoU information as a training signal. Our analysis demonstrated that zooming in on the main object improves classification accuracy. However, we observed that the model's performance suffers from data-related aspects, such as dataset bias towards a specific category.

Video Domain

In the video domain, we focused on spatiotemporal correspondence learning, a fundamental task that facilitates several applications such as VOS and MOT.

tracking smaller objects

In VOS, we performed an in-depth study on a state-of-the-art model that utilizes RNNs for effectively processing visual and motion cues for segmenting and tracking the target objects [Xu+18]. We identified that despite having a good accuracy score, the model's performance declines when handling smaller objects and segmenting fine object details around the edges. We argued that this is caused by information loss in the used encoder-decoder architecture due to several down-sampling operations. We proposed the skip-memory module that, by tracking objects at multiple feature resolutions, enhances the tracking of small objects. Additionally, we integrated a multi-task objective based on border distance classification that further improved the quality of the segmented object edges by incorporating additional location information into the model.

segmenting the object edges

tracking longer videos

Another limitation we observed in current VOS models pertains to their ability to process longer videos. This can occur due to several reasons. In longer videos, object appearance changes over time compared to the reference frame, making locating the target object more difficult. Moreover, due to model prediction inaccuracies, error accumulates over time, resulting in drift. On top of that, RNN-based models struggle with longer videos due to known RNN limitations such as vanishing gradient and catastrophic forgetting. This becomes particularly problematic when tracking occluded objects. To this end, we developed a hybrid VOS model that combines the merits of RNNs with template matching. For this hybrid strategy, we design a feature fusion module that allows the model to flexibly utilize information from the RNN- and matching-based branches. We experimentally showed this using the additional matching information significantly boosts the VOS performance, especially for longer videos and occluded objects. Moreover, we examined two architecture variants: bidirectional processing using information from past and future and multi-task learning with unsupervised optical flow objective, where we observed the bidirectional design further betters the VOS performance.

occlusion

annotation scarcity for videos

In the next part of this thesis, we discuss the challenges of spatiotemporal correspondence learning for MOT application, given the scarcity of annotated data. Several self-supervised methods have been recently proposed to facilitate correspondence learning on large amounts of unlabeled videos. However, these methods do not scale well to crowded scenes containing several similar objects, which is often the case in real-world tracking settings. To address this, we introduced S^{3}Track, a simple yet highly effective MOT framework that operates without using any video-level annotations. In S^{3}Track, the model is *directly optimized for the cross-frame object association*, using a soft matching layer implemented using

optimal transport and association pseudo labels obtained from motion information over short video clips. As a result, the model is devoid of alignment issues between the proxy objective and the downstream task, which can often pose a challenge in self-supervised learning solutions. We empirically demonstrated that the synchronization between the training objective and the inference task results in a strong appearance model in S^3Track that is robust in handling crowded scenes and occluded objects, achieving performance on par with supervised baselines.

crowded scenes, occlusion

In the last part of this thesis, we delved into a challenging setup that significantly affects the applicability of learning-based methods in real-world measurements: covariate shift between the distributions of training and testing data. Specifically, we focused on this crucial aspect within the context of video correspondence learning. In this regard, we took inspiration from recent TTA methods proposed to adjust trained neural networks to new unseen domains, using self-supervision without requiring labeled annotations. We introduced several strategies for adapting these methods to the video domain and conducted a comprehensive benchmark to demonstrate the efficacy of the proposed solutions across various domain shift scenarios.

domain adaptation using unlabeled videos

8.1 Future Work

In this thesis, we attempted to facilitate several shortcomings that hinder the reliable application of current deep-learning models in our daily lives. However, many challenges remain that demand further attention and exploration.

Dataset generation. In recent years, self-supervised learning approaches have achieved significant success in reducing the reliance on human-annotated data. However, supervised learning remains vital in a variety of applications. As a result, there is an urgent need for more extensive datasets that better reflect the complexities of real-world scenarios. Additionally, a promising research direction is to explore ways to effectively combine limited annotated data with a vast amount of unlabeled data in the context of **semi-supervised learning**.

Active and continual learning. Active learning is a machine learning technique that allows models to select the most informative data points to be labeled. This can help to improve the performance of the model by reducing the risk of overfitting to overrepresented data and ensuring that the model is exposed to a diverse range of data. Continual learning is a technique that allows machine learning models to learn new information without forgetting what they have already learned. This is a highly desired aspect for learning-based solutions, considering the dynamic nature of the world where the data is constantly changing. Both

these research areas are still in their early stages of development, but they have the potential to benefit computer vision significantly.

Multi-task learning. This research direction concerns the development of models that can perform multiple tasks sharing the same or the majority of the network parameters. As the final goal is to create machines that can visually perceive the world, it is not sensible to rely on several distinct neural networks, each specialized in a single task in an isolated manner. While previous works have proposed unified architectures with shared models to solve different tasks, specialized models still exhibit notably better performance. This presents an intriguing avenue for research: creating models that can effectively handle various tasks, leveraging the benefits of shared knowledge while preserving the advantages of task-specific expertise.

Along a similar note, **multi-modality learning** is another promising research direction that seeks to combine data from different modalities, such as images, text, or audio, to enhance overall performance and enable a more comprehensive understanding of sensory inputs. By exploring these research directions, the computer vision community can work towards developing more versatile and holistic models that go beyond single-task specialization, leading to more capable and intelligent vision systems with broader applications and improved perceptual capabilities.

Bibliography

[Alh+08] Faraj Alhwarin, Chao Wang, Danijela Ristić-Durrant, and Axel Gräser. "Improved SIFT-features matching for object recognition". In: *Visions of computer science-BCS international academic conference*. 2008, pp. 179–190 (cit. on p. 31).

[Azi+21a] Fatemeh Azimi, Benjamin Bischke, Sebastian Palacio, et al. "Revisiting Sequence-to-Sequence Video Object Segmentation with Multi-Task Loss and Skip-Memory". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE. 2021, pp. 5376–5383 (cit. on pp. 62, 81, 82, 92).

[Azi+23] Fatemeh Azimi, David Dembinsky, Federico Raue, et al. "Sequential Spatial Transformer Networks for Salient Object Classification". In: *12th International Conference on Pattern Recognition Applications and Methods (ICPRAM)*. SCITEPRESS, 2023 (cit. on p. 26).

[Azi+21b] Fatemeh Azimi, Stanislav Frolov, Federico Raue, Jörn Hees, and Andreas Dengel. "Hybrid-S2S: Video Object Segmentation with Recurrent Networks and Correspondence Matching." In: *VISIGRAPP (4: VISAPP)*. 2021, pp. 182–192 (cit. on pp. 62, 92).

[AMH23] Fatemeh Azimi, Fahim Mannan, and Felix Heide. "S3Track: Self-supervised Tracking with Soft Assignment Flow". In: *arXiv preprint arXiv:2305.09981* (2023) (cit. on p. 98).

[Azi+22a] Fatemeh Azimi, Jean-Francois Jacques Nicolas Nies, Sebastian Palacio, et al. "Spatial Transformer Networks for Curriculum Learning". In: *The International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. IEEE. 2022 (cit. on p. 26).

[Azi+22b] Fatemeh Azimi, Sebastian Palacio, Federico Raue, et al. "Self-supervised test-time adaptation on video data". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2022, pp. 3439–3448 (cit. on p. 98).

[Azi+19] Fatemeh Azimi, Federico Raue, Jörn Hees, and Andreas Dengel. "A Reinforcement Learning Approach for Sequential Spatial Transformer Networks". In: *International Conference on Artificial Neural Networks*. Springer. 2019, pp. 585–597 (cit. on pp. 26, 43, 44, 46, 48–51).

[Azi+21c] Fatemeh Azimi, Federico Raue, Jörn Hees, and Andreas Dengel. "Rethinking RNN-based Video Object Segmentation". In: *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISIGRAPP 2021*. Springer. 2021 (cit. on p. 62).

[BMK14] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. "Multiple object recognition with visual attention". In: *arXiv preprint arXiv:1412.7755* (2014) (cit. on p. 32).

[BKC17] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. “Segnet: A deep convolutional encoder-decoder architecture for image segmentation”. In: *IEEE transactions on pattern analysis and machine intelligence* 39.12 (2017), pp. 2481–2495 (cit. on pp. 66, 68).

[Bah+16] Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, et al. “An actor-critic algorithm for sequence prediction”. In: *arXiv preprint arXiv:1607.07086* (2016) (cit. on pp. 16, 17, 40).

[BCB14] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. “Neural machine translation by jointly learning to align and translate”. In: *arXiv preprint arXiv:1409.0473* (2014) (cit. on pp. 76, 80).

[BSC18] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. “Metareg: Towards domain generalization using meta-regularization”. In: *Advances in Neural Information Processing Systems* 31 (2018), pp. 998–1008 (cit. on p. 102).

[BWL18] Linchao Bao, Baoyuan Wu, and Wei Liu. “CNN in MRF: Video object segmentation via inference in a CNN-based higher-order spatio-temporal MRF”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 5977–5986 (cit. on p. 64).

[BHM21] Favyen Bastani, Songtao He, and Samuel Madden. “Self-Supervised Multi-Object Tracking with Cross-Input Consistency”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 13695–13706 (cit. on pp. 102, 110, 114).

[Bay+08] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. “Speeded-up robust features (SURF)”. In: *Computer vision and image understanding* 110.3 (2008), pp. 346–359 (cit. on p. 31).

[Ben+15] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. “Scheduled sampling for sequence prediction with recurrent neural networks”. In: *arXiv preprint arXiv:1506.03099* (2015) (cit. on pp. 33, 81).

[Ben+09] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. “Curriculum learning”. In: *Proceedings of the 26th annual international conference on machine learning*. 2009, pp. 41–48 (cit. on pp. 32, 43).

[BML19a] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. “Tracking without bells and whistles”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 941–951 (cit. on p. 101).

[BML19b] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixé. “Tracking Without Bells and Whistles”. In: *The IEEE International Conference on Computer Vision (ICCV)*. Nov. 2019 (cit. on p. 101).

[BS08] Keni Bernardin and Rainer Stiefelhagen. “Evaluating multiple object tracking performance: the clear mot metrics”. In: *EURASIP Journal on Image and Video Processing* 2008 (2008), pp. 1–10 (cit. on p. 114).

[Bew+16] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. “Simple online and realtime tracking”. In: *2016 IEEE international conference on image processing (ICIP)*. IEEE. 2016, pp. 3464–3468 (cit. on pp. 101, 114, 116).

[Bha+20] Goutam Bhat, Felix Järemo Lawin, Martin Danelljan, et al. "Learning what to learn for video object segmentation". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer. 2020, pp. 777–794 (cit. on p. 65).

[Bia+22] Zhangxing Bian, Allan Jabri, Alexei A Efros, and Andrew Owens. "Learning pixel trajectories with multiscale contrastive random walks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 6508–6519 (cit. on p. 100).

[Bis+19] Benjamin Bischke, Patrick Helber, Joachim Folz, Damian Borth, and Andreas Dengel. "Multi-task learning for segmentation of building footprints with deep neural networks". In: *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2019, pp. 1480–1484 (cit. on p. 69).

[BES17] Erik Bochinski, Volker Eiselein, and Thomas Sikora. "High-speed tracking-by-detection without using image information". In: *2017 14th IEEE international conference on advanced video and signal based surveillance (AVSS)*. IEEE. 2017, pp. 1–6 (cit. on pp. 101, 114, 116).

[Bom+21] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, et al. "On the opportunities and risks of foundation models". In: *arXiv preprint arXiv:2108.07258* (2021) (cit. on p. 120).

[BSI13] Ali Borji, Dicky N Sihite, and Laurent Itti. "What stands out in a scene? A study of human explicit saliency judgment". In: *Vision research* 91 (2013), pp. 62–77 (cit. on p. 6).

[BL20] Guillem Brasó and Laura Leal-Taixé. "Learning a neural solver for multiple object tracking". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 6247–6257 (cit. on p. 101).

[Bre+09] Michael D Breitenstein, Fabian Reichlin, Bastian Leibe, Esther Koller-Meier, and Luc Van Gool. "Robust tracking-by-detection using a detector confidence particle filter". In: *2009 IEEE 12th International Conference on Computer Vision*. IEEE. 2009, pp. 1515–1522 (cit. on p. 104).

[BL02] Matthew Brown and David G Lowe. "Invariant features from interest point groups." In: *Bmvc*. Vol. 4. 2002, pp. 398–410 (cit. on p. 31).

[Bro+20] Tom Brown, Benjamin Mann, Nick Ryder, et al. "Language models are few-shot learners". In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901 (cit. on p. 2).

[BM10] Thomas Brox and Jitendra Malik. "Object segmentation by long term analysis of point trajectories". In: *European conference on computer vision*. Springer. 2010, pp. 282–295 (cit. on p. 63).

[BS21] Collin Burns and Jacob Steinhardt. "Limitations of Post-Hoc Feature Alignment for Robustness". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 2525–2533 (cit. on p. 103).

[Cae+17] Sergi Caelles, Kevis-Kokitsi Maninis, Jordi Pont-Tuset, et al. “One-shot video object segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 221–230 (cit. on pp. 69, 73, 120).

[Cae+20] Holger Caesar, Varun Bankiti, Alex H Lang, et al. “nuscenes: A multimodal dataset for autonomous driving”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 11621–11631 (cit. on pp. 104, 105, 113, 115, 116).

[Cai+22] Jiarui Cai, Mingze Xu, Wei Li, et al. “MeMOT: Multi-Object Tracking with Memory”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 8090–8100 (cit. on p. 101).

[Cal+10] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. “Brief: Binary robust independent elementary features”. In: *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*. Springer. 2010, pp. 778–792 (cit. on p. 31).

[Cao+15] Chunshui Cao, Xianming Liu, Yi Yang, et al. “Look and think twice: Capturing top-down visual attention with feedback convolutional neural networks”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2956–2964 (cit. on p. 32).

[Cao+22] Jinkun Cao, Xinshuo Weng, Rawal Khirodkar, Jiangmiao Pang, and Kris Kitani. “Observation-Centric SORT: Rethinking SORT for Robust Multi-Object Tracking”. In: *arXiv preprint arXiv:2203.14360* (2022) (cit. on pp. 101, 114–116).

[Car+20] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, et al. “End-to-end object detection with transformers”. In: *European conference on computer vision*. Springer. 2020, pp. 213–229 (cit. on pp. 101, 104).

[Car+19] Fabio M Carlucci, Antonio D’Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. “Domain generalization by solving jigsaw puzzles”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2229–2238 (cit. on p. 102).

[Car+17] Fabio Maria Carlucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulo. “Autodial: Automatic domain alignment layers”. In: *2017 IEEE international conference on computer vision (ICCV)*. IEEE. 2017, pp. 5077–5085 (cit. on p. 102).

[Cha+21] Mohamed Chaabane, Peter Zhang, J Ross Beveridge, and Stephen O’Hara. “Deft: Detection embeddings for tracking”. In: *arXiv preprint arXiv:2102.02267* (2021) (cit. on pp. 114, 115).

[CLM17] Haw-Shiuan Chang, Erik Learned-Miller, and Andrew McCallum. “Active bias: Training more accurate neural networks by emphasizing high variance samples”. In: *arXiv preprint arXiv:1704.07433* (2017) (cit. on p. 43).

[Che+17a] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs”. In: *IEEE transactions on pattern analysis and machine intelligence* 40.4 (2017), pp. 834–848 (cit. on pp. 32, 64).

[Che+17b] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. "Rethinking atrous convolution for semantic image segmentation". In: *arXiv preprint arXiv:1706.05587* (2017) (cit. on pp. 93, 94).

[Che+20a] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. "A simple framework for contrastive learning of visual representations". In: *International conference on machine learning*. PMLR. 2020, pp. 1597–1607 (cit. on p. 99).

[Che+20b] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. "Improved baselines with momentum contrastive learning". In: *arXiv preprint arXiv:2003.04297* (2020) (cit. on p. 99).

[Che+18] Yuhua Chen, Jordi Pont-Tuset, Alberto Montes, and Luc Van Gool. "Blazingly fast video object segmentation with pixel-wise metric learning". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 1189–1198 (cit. on p. 65).

[CTT21] Ho Kei Cheng, Yu-Wing Tai, and Chi-Keung Tang. "Rethinking space-time networks with improved memory coverage for efficient video object segmentation". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 11781–11794 (cit. on p. 65).

[CL19] Peng Chu and Haibin Ling. "Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 6172–6181 (cit. on p. 114).

[Cia+20] Gioele Ciaparrone, Francisco Luque Sánchez, Siham Tabik, et al. "Deep learning in video multi-object tracking: A survey". In: *Neurocomputing* 381 (2020), pp. 61–88 (cit. on p. 103).

[Cui+21] Jiequan Cui, Zhisheng Zhong, Shu Liu, Bei Yu, and Jiaya Jia. "Parametric contrastive learning". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 715–724 (cit. on p. 99).

[Cut13] Marco Cuturi. "Sinkhorn distances: Lightspeed computation of optimal transport". In: *Advances in neural information processing systems* 26 (2013) (cit. on pp. 106, 107).

[Dai+17] Jifeng Dai, Haozhi Qi, Yuwen Xiong, et al. "Deformable convolutional networks". In: *Proceedings of the IEEE international conference on computer vision*. doi: 10.1109/ICCV.2017.89. 2017, pp. 764–773 (cit. on p. 31).

[Dav+20] Achal Dave, Tarasha Khurana, Pavel Tokmakov, Cordelia Schmid, and Deva Ramanan. "TAO: A Large-Scale Benchmark for Tracking Any Object". In: *arXiv preprint arXiv:2005.10356* (2020) (cit. on pp. 120, 126).

[Den+09] Jia Deng, Wei Dong, Richard Socher, et al. "Imagenet: A large-scale hierarchical image database". In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. doi: 10.1109/cvprw.2009.5206848. Ieee. 2009, pp. 248–255 (cit. on pp. 1, 3, 25, 31, 50, 63, 81).

[DGE15] Carl Doersch, Abhinav Gupta, and Alexei A Efros. "Unsupervised visual representation learning by context prediction". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1422–1430 (cit. on p. 99).

[DGZ17] Qi Dong, Shaogang Gong, and Xiatian Zhu. "Class rectification hard mining for imbalanced deep learning". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 1851–1860 (cit. on p. 55).

[Dos+20] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *CoRR* abs/2010.11929 (2020). arXiv: 2010.11929 (cit. on pp. 20, 50, 65, 101).

[Duk+21] Brendan Duke, Abdalla Ahmed, Christian Wolf, Parham Aarabi, and Graham W Taylor. "Sstvos: Sparse spatiotemporal transformers for video object segmentation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 5912–5921 (cit. on p. 65).

[Ebe+17] Frederik Ebert, Chelsea Finn, Alex X Lee, and Sergey Levine. "Self-supervised visual planning with temporal skip connections". In: *arXiv preprint arXiv:1710.05268* (2017) (cit. on p. 79).

[Elm93] Jeffrey L Elman. "Learning and development in neural networks: The importance of starting small". In: *Cognition* 48.1 (1993), pp. 71–99 (cit. on p. 43).

[Eve+10] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. "The pascal visual object classes (voc) challenge". In: *International journal of computer vision* 88.2 (2010), pp. 303–338 (cit. on pp. 50, 53, 54).

[FI14] Alon Faktor and Michal Irani. "Video Segmentation by Non-Local Consensus voting." In: *BMVC*. 2014, p. 8 (cit. on p. 63).

[Fan+18] Yang Fan, Fei Tian, Tao Qin, Xiang-Yang Li, and Tie-Yan Liu. "Learning to teach". In: *arXiv preprint arXiv:1805.03643* (2018) (cit. on p. 33).

[Flo+17] Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. "Reverse curriculum generation for reinforcement learning". In: *Conference on robot learning*. PMLR. 2017, pp. 482–495 (cit. on p. 33).

[FS22] Masato Fujitake and Akihiro Sugimoto. "Video Sparse Transformer With Attention-Guided Memory for Video Object Detection". In: *IEEE Access* 10 (2022), pp. 65886–65900 (cit. on p. 65).

[Gao+22] Mingqi Gao, Feng Zheng, James JQ Yu, et al. "Deep learning for video object segmentation: a review". In: *Artificial Intelligence Review* (2022), pp. 1–75 (cit. on p. 61).

[GLU12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012 (cit. on pp. 103, 105, 111, 112, 114, 115, 117, 118).

[GSK18] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. "Unsupervised representation learning by predicting image rotations". In: *arXiv preprint arXiv:1803.07728* (2018) (cit. on p. 99).

[GB10] Xavier Glorot and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 2010, pp. 249–256 (cit. on pp. 71, 72).

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. `http://www.deeplearningbook.org`. MIT Press, 2016 (cit. on pp. 12, 22).

[GB+05] Yves Grandvalet, Yoshua Bengio, et al. "Semi-supervised learning by entropy minimization." In: *CAP* 367 (2005), pp. 281–296 (cit. on p. 103).

[Gra+17] Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. "Automated curriculum learning for neural networks". In: *international conference on machine learning*. PMLR. 2017, pp. 1311–1320 (cit. on p. 33).

[GFS05] Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. "Bidirectional LSTM networks for improved phoneme classification and recognition". In: *International conference on artificial neural networks*. Springer. 2005, pp. 799–804 (cit. on p. 88).

[Gra+16] Alex Graves, Greg Wayne, Malcolm Reynolds, et al. "Hybrid computing using a neural network with dynamic external memory". In: *Nature* 538.7626 (2016), pp. 471–476 (cit. on p. 33).

[Gre+15] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. "Draw: A recurrent neural network for image generation". In: *arXiv preprint arXiv:1502.04623* (2015) (cit. on p. 38).

[Gui+23] Jie Gui, Tuo Chen, Qiong Cao, et al. "A Survey of Self-Supervised Learning from Multiple Perspectives: Algorithms, Theory, Applications and Future Trends". In: *arXiv preprint arXiv:2301.05712* (2023) (cit. on p. 97).

[Guo+21] Song Guo, Jingya Wang, Xinchao Wang, and Dacheng Tao. "Online multiple object tracking with cross-task synergy". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 8136–8145 (cit. on p. 103).

[HW19] Guy Hacohen and Daphna Weinshall. "On the power of curriculum learning in training deep networks". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 2535–2544 (cit. on pp. 33, 43).

[HS+88] Chris Harris, Mike Stephens, et al. "A combined corner and edge detector". In: *Alvey vision conference*. Vol. 15. 50. Citeseer. 1988, pp. 10–5244 (cit. on p. 31).

[Has+22] Mohammed Hassanin, Saeed Anwar, Ibrahim Radwan, Fahad S Khan, and Ajmal Mian. "Visual attention methods in deep learning: An in-depth survey". In: *arXiv preprint arXiv:2204.07756* (2022) (cit. on p. 32).

[He+21] Jiawei He, Zehao Huang, Naiyan Wang, and Zhaoxiang Zhang. "Learnable graph matching: Incorporating graph partitioning with deep feature learning for multiple object tracking". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 5299–5309 (cit. on pp. 101, 103).

[He+20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. “Momentum contrast for unsupervised visual representation learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 9729–9738 (cit. on p. 99).

[He+17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. “Mask r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2961–2969 (cit. on p. 63).

[He+16] 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*. doi:10.1109/CVPR.2016.90. 2016, pp. 770–778 (cit. on pp. 20, 22, 25, 31, 50, 54, 63, 81, 93, 94, 110, 111).

[He+15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Spatial pyramid pooling in deep convolutional networks for visual recognition”. In: *IEEE transactions on pattern analysis and machine intelligence* 37.9 (2015), pp. 1904–1916 (cit. on p. 104).

[HD19a] Dan Hendrycks and Thomas Dietterich. “Benchmarking Neural Network Robustness to Common Corruptions and Perturbations”. In: *Proceedings of the International Conference on Learning Representations* (2019) (cit. on p. 127).

[HD19b] Dan Hendrycks and Thomas Dietterich. “Benchmarking neural network robustness to common corruptions and perturbations”. In: *iclr*. 2019 (cit. on pp. 50, 120, 122, 127).

[Hen+21] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. “Natural Adversarial Examples”. In: *CVPR* (2021) (cit. on p. 25).

[HVD15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. “Distilling the knowledge in a neural network”. In: *arXiv preprint arXiv:1503.02531* (2015) (cit. on p. 33).

[Hin+12] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. “Improving neural networks by preventing co-adaptation of feature detectors”. In: *arXiv preprint arXiv:1207.0580* (2012) (cit. on p. 19).

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851 (cit. on p. 2).

[Ho+19] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. “Axial attention in multidimensional transformers”. In: *arXiv preprint arXiv:1912.12180* (2019) (cit. on pp. 17, 93).

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780 (cit. on pp. 15, 36, 46).

[Hon+22] Dexiang Hong, Guorong Li, Bineng Zhong, et al. “CRNet: Collaborative Refinement Network for Self-Supervised Video Object Segmentation”. In: *2022 IEEE 5th International Conference on Multimedia Information Processing and Retrieval (MIPR)*. IEEE. 2022, pp. 172–177 (cit. on p. 100).

[Hu+21] Li Hu, Peng Zhang, Bang Zhang, et al. “Learning position and target consistency for memory-based video object segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 4144–4154 (cit. on p. 65).

[Hu+18] Ping Hu, Gang Wang, Xiangfei Kong, Jason Kuen, and Yap-Peng Tan. “Motion-guided cascaded refinement network for video object segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 1400–1409 (cit. on p. 64).

[HHS18] Yuan-Ting Hu, Jia-Bin Huang, and Alexander G Schwing. “Videomatch: Matching based video object segmentation”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 54–70 (cit. on p. 65).

[Hua+17] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. “Densely connected convolutional networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. doi: 10.1109/cvpr.2017.243. 2017, pp. 4700–4708 (cit. on p. 31).

[HHK18] Haoshuo Huang, Qixing Huang, and Philipp Krahenbuhl. “Domain transfer through deep activation matching”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 590–605 (cit. on p. 102).

[Hyu+22] Jeongseok Hyun, Myunggu Kang, Dongyoon Wee, and Dit-Yan Yeung. “Detection Recovery in Online Multi-Object Tracking with Sparse Graph Tracker”. In: *arXiv preprint arXiv:2205.00968* (2022) (cit. on p. 101).

[IS15] Sergey Ioffe and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *International conference on machine learning*. PMLR. 2015, pp. 448–456 (cit. on pp. 19, 103).

[JOE20] Allan Jabri, Andrew Owens, and Alexei A Efros. “Space-time correspondence as a contrastive random walk”. In: *arXiv preprint arXiv:2006.14613* (2020) (cit. on pp. 97, 99, 100, 120, 122, 124–126, 128, 130).

[JSZ+15] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. “Spatial transformer networks”. In: *Advances in neural information processing systems*. 2015, pp. 2017–2025 (cit. on pp. 25, 32, 34, 35, 37, 39, 43, 49, 50).

[JG14] Suyog Dutt Jain and Kristen Grauman. “Supervoxel-consistent foreground propagation in video”. In: *European conference on computer vision*. Springer. 2014, pp. 656–671 (cit. on p. 63).

[Jai+21] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. “A survey on contrastive self-supervised learning”. In: *Technologies* (2021) (cit. on p. 97).

[Jeo+21] Sangryul Jeon, Dongbo Min, Seungryong Kim, and Kwanghoon Sohn. “Mining better samples for contrastive learning of temporal correspondence”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 1034–1044 (cit. on p. 100).

[JK17] Yunho Jeon and Junmo Kim. “Active convolution: Learning the shape of convolution for image classification”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4201–4209 (cit. on p. 31).

[Jia+16] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. “Dynamic filter networks”. In: *Advances in neural information processing systems* 29 (2016) (cit. on p. 31).

[Jia+18] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. “Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels”. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 2304–2313 (cit. on p. 33).

[Jia+19] Licheng Jiao, Fan Zhang, Fang Liu, et al. “A survey of deep learning-based object detection”. In: *IEEE access* 7 (2019), pp. 128837–128868 (cit. on p. 104).

[JT20] Longlong Jing and Yingli Tian. “Self-supervised visual feature learning with deep neural networks: A survey”. In: *IEEE transactions on pattern analysis and machine intelligence* (2020) (cit. on p. 99).

[Joh+19] Joakim Johnander, Martin Danelljan, Emil Brissman, Fahad Shahbaz Khan, and Michael Felsberg. “A generative appearance model for end-to-end video object segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 8953–8962 (cit. on pp. 64, 65, 92).

[Jon+20] Rico Jonschkowski, Austin Stone, Jonathan T Barron, et al. “What matters in unsupervised optical flow”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer. 2020, pp. 557–572 (cit. on p. 90).

[LHS20] Phuc H Le-Khac, Graham Healy, and Alan F Smeaton. “Contrastive representation learning: A framework and review”. In: *Ieee Access* 8 (2020), pp. 193907–193934 (cit. on p. 99).

[KOL21] Aleksandr Kim, Aljoša Ošep, and Laura Leal-Taixé. “Eagermot: 3d multi-object tracking via sensor fusion”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 11315–11321 (cit. on pp. 114, 115).

[Kim+18] Dahun Kim, Donghyeon Cho, Donggeun Yoo, and In So Kweon. “Learning image representations by completing damaged jigsaw puzzles”. In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2018, pp. 793–802 (cit. on p. 99).

[KB14] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014) (cit. on pp. 23, 24, 46, 54, 72, 81).

[KF19] Shu Kong and Charless Fowlkes. “Multigrid predictive filter flow for unsupervised learning on videos”. In: *arXiv preprint arXiv:1904.01693* (2019) (cit. on p. 100).

[KK11] Philipp Krähenbühl and Vladlen Koltun. “Efficient inference in fully connected crfs with gaussian edge potentials”. In: *Advances in neural information processing systems*. 2011, pp. 109–117 (cit. on p. 72).

[KH+09] Alex Krizhevsky, Geoffrey Hinton, et al. “Learning multiple layers of features from tiny images”. In: (2009) (cit. on pp. 3, 25, 50).

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105 (cit. on pp. 1, 20, 31, 50, 63, 71, 72).

[KH91] Anders Krogh and John Hertz. “A simple weight decay can improve generalization”. In: *Advances in neural information processing systems* 4 (1991) (cit. on p. 18).

[KD09] Kai A Krueger and Peter Dayan. “Flexible shaping: How learning in small steps helps”. In: *Cognition* 110.3 (2009), pp. 380–394 (cit. on p. 32).

[Kum+18] Abhishek Kumar, Prasanna Sattigeri, Kahini Wadhawan, et al. “Co-regularized alignment for unsupervised domain adaptation”. In: *arXiv preprint arXiv:1811.05443* (2018) (cit. on p. 102).

[LX19a] Z. Lai and W. Xie. “Self-supervised Learning for Video Correspondence Flow”. In: *BMVC*. 2019 (cit. on pp. 99, 100).

[LLX20] Zihang Lai, Erika Lu, and Weidi Xie. “Mast: A memory-augmented self-supervised tracker”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 6479–6488 (cit. on pp. 97, 100, 120, 122, 124, 125, 128, 130).

[LX19b] Zihang Lai and Weidi Xie. “Self-supervised learning for video correspondence flow”. In: *arXiv preprint arXiv:1905.00875* (2019) (cit. on p. 124).

[LMS17] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. “Colorization as a proxy task for visual understanding”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 6874–6883 (cit. on p. 99).

[LeC+98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998). doi: 10.1109/9780470544976.ch9, pp. 2278–2324 (cit. on p. 31).

[Lev86] Hector J Levesque. “Knowledge representation and reasoning”. In: *Annual review of computer science* 1.1 (1986), pp. 255–287 (cit. on p. 1).

[Lev21] Sergey Levine. *Deep Reinforcement Learning*. Sept. 2021 (cit. on pp. 28–31).

[Li20] Fei-Fei Li. *Deep Learning for Computer Vision*. Sept. 2020 (cit. on pp. 12, 15).

[Li+22a] Guorong Li, Dexiang Hong, Kai Xu, et al. “Self Supervised Progressive Network for High Performance Video Object Segmentation”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2022) (cit. on p. 100).

[LGJ20] Jiahe Li, Xu Gao, and Tingting Jiang. “Graph networks for multiple object tracking”. In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2020, pp. 719–728 (cit. on p. 101).

[Li+22b] Liulei Li, Tianfei Zhou, Wenguan Wang, et al. “Locality-aware inter-and intra-video reconstruction for self-supervised correspondence learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 8719–8730 (cit. on p. 100).

[Li+22c] Mingxing Li, Li Hu, Zhiwei Xiong, et al. “Recurrent dynamic embedding for video object segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 1332–1341 (cit. on p. 65).

[Li+22d] Ruoqi Li, Yifan Wang, Lijun Wang, et al. “From Pixels to Semantics: Self-Supervised Video Object Segmentation With Multiperspective Feature Mining”. In: *IEEE Transactions on Image Processing* 31 (2022), pp. 5801–5812 (cit. on p. 100).

[LKR22] Shuai Li, Yu Kong, and Hamid Rezatofighi. “Learning of Global Objective for Network Flow in Multi-Object Tracking”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 8855–8865 (cit. on p. 103).

[Li+17] Xiaoxiao Li, Yuankai Qi, Zhe Wang, et al. “Video object segmentation with re-identification”. In: *arXiv preprint arXiv:1708.00197* (2017) (cit. on p. 63).

[Li+19a] Xueting Li, Sifei Liu, Shalini De Mello, et al. “Joint-task self-supervised learning for temporal correspondence”. In: *arXiv preprint arXiv:1909.11895* (2019) (cit. on p. 100).

[Li+16] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. “Revisiting batch normalization for practical domain adaptation”. In: *arXiv preprint arXiv:1603.04779* (2016) (cit. on p. 103).

[Li+19b] Yiying Li, Yongxin Yang, Wei Zhou, and Timothy Hospedales. “Feature-critic networks for heterogeneous domain generalization”. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 3915–3924 (cit. on p. 102).

[LHF20] Jian Liang, Dapeng Hu, and Jiashi Feng. “Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 6028–6039 (cit. on p. 102).

[LL17] Chen-Hsuan Lin and Simon Lucey. “Inverse compositional spatial transformer networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 2568–2576 (cit. on p. 32).

[Lin+17] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, et al. “Feature pyramid networks for object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2117–2125 (cit. on pp. 22, 104, 110, 111).

[Lin+14] Tsung-Yi Lin, Michael Maire, Serge Belongie, et al. “Microsoft coco: Common objects in context”. In: *European conference on computer vision*. Springer. 2014, pp. 740–755 (cit. on pp. 111, 116, 118).

[Lin+22] Zhihui Lin, Tianyu Yang, Maomao Li, et al. “SWEM: Towards Real-Time Video Object Segmentation with Sequential Weighted Expectation-Maximization”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 1362–1372 (cit. on p. 65).

[Liu+21] Ze Liu, Yutong Lin, Yue Cao, et al. “Swin transformer: Hierarchical vision transformer using shifted windows”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 10012–10022 (cit. on p. 104).

[LSD15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440 (cit. on p. 71).

[Low04] David G Lowe. “Distinctive image features from scale-invariant keypoints”. In: *International journal of computer vision* 60.2 (2004). doi: 10.1023/B:VISI.0000029664.99615.94, pp. 91–110 (cit. on pp. 22, 31).

[Low99] David G Lowe. “Object recognition from local scale-invariant features”. In: *Proceedings of the seventh IEEE international conference on computer vision*. Vol. 2. Ieee. 1999, pp. 1150–1157 (cit. on p. 31).

[Low87] David G Lowe. “The viewpoint consistency constraint”. In: *International Journal of Computer Vision* 1.1 (1987), pp. 57–72 (cit. on p. 31).

[Lug05] George F Luger. *Artificial intelligence: structures and strategies for complex problem solving*. Pearson education, 2005 (cit. on p. 1).

[Lui+21] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, et al. “Hota: A higher order metric for evaluating multi-object tracking”. In: *International journal of computer vision* 129.2 (2021), pp. 548–578 (cit. on p. 114).

[LVL18] Jonathon Luiten, Paul Voigtlaender, and Bastian Leibe. “PReMVOS: Proposal-generation, refinement and merging for video object segmentation”. In: *Asian Conference on Computer Vision*. Springer. 2018, pp. 565–580 (cit. on p. 63).

[Luo+16] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. “Understanding the effective receptive field in deep convolutional neural networks”. In: *Advances in neural information processing systems* 29 (2016) (cit. on p. 32).

[Man+18] Kevis-Kokitsi Maninis, Sergi Caelles, Yuhua Chen, et al. “Video Object Segmentation Without Temporal Information”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (2018) (cit. on pp. 63, 82, 92).

[Man+16] Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Pablo Arbeláez, and Luc Van Gool. “Deep retinal image understanding”. In: *International conference on medical image computing and computer-assisted intervention*. Springer. 2016, pp. 140–148 (cit. on p. 63).

[Mat+19] Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. “Teacher–student curriculum learning”. In: *IEEE transactions on neural networks and learning systems* 31.9 (2019), pp. 3732–3740 (cit. on p. 33).

[McC+06] John McCarthy, Marvin L Minsky, Nathaniel Rochester, and Claude E Shannon. “A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955”. In: *AI magazine* 27.4 (2006), pp. 12–12 (cit. on p. 1).

[MC04] Pamela McCorduck and Cli Cfe. *Machines who think: A personal inquiry into the history and prospects of artificial intelligence*. CRC Press, 2004 (cit. on p. 1).

[Mei+22] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. "Trackformer: Multi-object tracking with transformers". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 8844–8854 (cit. on p. 101).

[MG18] Agnieszka Mikołajczyk and Michał Grochowski. "Data augmentation for improving deep learning in image classification problem". In: *2018 international interdisciplinary PhD workshop (IIPhDW)*. IEEE. 2018, pp. 117–122 (cit. on p. 31).

[Mik02] Krystian Mikolajczyk. "Detection of local features invariant to affines transformations". PhD thesis. Institut National Polytechnique de Grenoble-INPG, 2002 (cit. on p. 31).

[MHG+14] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. "Recurrent models of visual attention". In: *Advances in neural information processing systems*. 2014, pp. 2204–2212 (cit. on pp. 32, 38).

[Mni+13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, et al. "Playing atari with deep reinforcement learning". In: *arXiv preprint arXiv:1312.5602* (2013) (cit. on pp. 30, 50, 51, 53).

[Moo06] James Moor. "The Dartmouth College artificial intelligence conference: The next fifty years". In: *Ai Magazine* 27.4 (2006), pp. 87–87 (cit. on p. 1).

[MBS13] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. "Domain generalization via invariant feature representation". In: *International Conference on Machine Learning*. PMLR. 2013, pp. 10–18 (cit. on p. 102).

[MHC18] T Nathan Mundhenk, Daniel Ho, and Barry Y Chen. "Improvements to context based self-supervised learning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 9339–9348 (cit. on p. 99).

[Mun57] James Munkres. "Algorithms for the assignment and transportation problems". In: *Journal of the society for industrial and applied mathematics* 5.1 (1957), pp. 32–38 (cit. on p. 106).

[Nad+20] Zachary Nado, Shreyas Padhy, D Sculley, et al. "Evaluating prediction-time batch normalization for robustness under covariate shift". In: *arXiv preprint arXiv:2006.10963* (2020) (cit. on pp. 103, 121–123).

[Nil82] Nils J Nilsson. *Principles of artificial intelligence*. Springer Science & Business Media, 1982 (cit. on p. 1).

[Nil10] Nils J. Nilsson. *The Quest for Artificial Intelligence: A History of Ideas and Achievements*. Cambridge University Press, 2010 (cit. on p. 1).

[NF16] Mehdi Noroozi and Paolo Favaro. "Unsupervised learning of visual representations by solving jigsaw puzzles". In: *European conference on computer vision*. Springer. 2016, pp. 69–84 (cit. on pp. 99, 102).

[Oh+19] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. "Video object segmentation using space-time memory networks". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 9226–9235 (cit. on pp. 65, 66, 82, 92).

[OLV18] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. "Representation learning with contrastive predictive coding". In: *arXiv preprint arXiv:1807.03748* (2018) (cit. on p. 99).

[Pan+19] Bo Pang, Kaiwen Zha, Hanwen Cao, Chen Shi, and Cewu Lu. "Deep rnn framework for visual sequential applications". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 423–432 (cit. on pp. 91–93).

[PF13] Anestis Papazoglou and Vittorio Ferrari. "Fast object segmentation in unconstrained video". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2013, pp. 1777–1784 (cit. on p. 63).

[Pas+17] Adam Paszke, Sam Gross, Soumith Chintala, et al. "Automatic differentiation in PyTorch". In: (2017) (cit. on p. 54).

[Pav10] P Ivan Pavlov. "Conditioned reflexes: an investigation of the physiological activity of the cerebral cortex". In: *Annals of neurosciences* 17.3 (2010), p. 136 (cit. on p. 32).

[Pen+17] Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. "Large kernel matters–improve semantic segmentation by global convolutional network". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4353–4361 (cit. on pp. 78, 81, 93).

[Pen+19] Xingchao Peng, Qinxun Bai, Xide Xia, et al. "Moment matching for multi-source domain adaptation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 1406–1415 (cit. on p. 102).

[Per+16a] F. Perazzi, J. Pont-Tuset, B. McWilliams, et al. "A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation". In: *Computer Vision and Pattern Recognition*. 2016 (cit. on p. 63).

[Per+17] 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*. 2017, pp. 2663–2672 (cit. on pp. 63, 64, 67, 73, 79, 82, 92).

[Per+16b] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, et al. "A benchmark dataset and evaluation methodology for video object segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 724–732 (cit. on pp. 71, 80).

[PW17] Luis Perez and Jason Wang. "The effectiveness of data augmentation in image classification using deep learning". In: *arXiv preprint arXiv:1712.04621* (2017) (cit. on p. 31).

[Pet04] Gail B Peterson. "A day of great illumination: BF Skinner's discovery of shaping". In: *Journal of the experimental analysis of behavior* 82.3 (2004), pp. 317–328 (cit. on pp. 32, 43).

[Pon+17] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, et al. "The 2017 davis challenge on video object segmentation". In: *arXiv preprint arXiv:1704.00675* (2017) (cit. on pp. 4, 81, 83, 120, 126).

[Rad+21] Alec Radford, Jong Wook Kim, Chris Hallacy, et al. “Learning transferable visual models from natural language supervision”. In: *International conference on machine learning*. PMLR. 2021, pp. 8748–8763 (cit. on p. 2).

[Rak+21] Lionel Rakai, Huansheng Song, ShiJie Sun, Wentao Zhang, and Yanni Yang. “Data association in multiple object tracking: A survey of recent techniques”. In: *Expert Systems with Applications* (2021), p. 116300 (cit. on p. 103).

[Rec+19] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. “Do imagenet classifiers generalize to imagenet?” In: *International Conference on Machine Learning*. PMLR. 2019, pp. 5389–5400 (cit. on p. 120).

[Red+18] Fitsum A Reda, Guilin Liu, Kevin J Shih, et al. “Sdc-net: Video prediction using spatially-displaced convolution”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 718–733 (cit. on p. 69).

[RM99] Russell Reed and Robert J MarksII. *Neural smithing: supervised learning in feedforward artificial neural networks*. Mit Press, 1999 (cit. on p. 18).

[Ren+15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. “Faster r-cnn: Towards real-time object detection with region proposal networks”. In: *Advances in neural information processing systems*. doi: 10.1109/TPAMI.2016.2577031. 2015, pp. 91–99 (cit. on pp. 63, 101, 104, 110, 112).

[Ris+16] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. “Performance measures and a data set for multi-target, multi-camera tracking”. In: *European conference on computer vision*. Springer. 2016, pp. 17–35 (cit. on p. 114).

[RP99] Douglas LT Rohde and David C Plaut. “Language acquisition in the absence of explicit negative evidence: How important is starting small?” In: *Cognition* 72.1 (1999), pp. 67–109 (cit. on p. 43).

[RFB15] 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*. Springer. 2015, pp. 234–241 (cit. on pp. 21, 68, 81).

[RD06] Edward Rosten and Tom Drummond. “Machine learning for high-speed corner detection”. In: *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9*. Springer. 2006, pp. 430–443 (cit. on p. 31).

[Roy+19] Subhankar Roy, Aliaksandr Siarohin, Enver Sangineto, et al. “Unsupervised domain adaptation using feature-whitening and consensus loss”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 9471–9480 (cit. on p. 103).

[Rub+11] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. “ORB: An efficient alternative to SIFT or SURF”. In: *2011 International conference on computer vision*. Ieee. 2011, pp. 2564–2571 (cit. on p. 31).

[Rud17] Sebastian Ruder. “An overview of multi-task learning in deep neural networks”. In: *arXiv preprint arXiv:1706.05098* (2017) (cit. on p. 89).

[Sai+19] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. "Semi-supervised domain adaptation via minimax entropy". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 8050–8058 (cit. on p. 103).

[San94] Terence D Sanger. "Neural network learning control of robot manipulators using gradually increasing task difficulty". In: *IEEE transactions on Robotics and Automation* 10.3 (1994), pp. 323–333 (cit. on p. 43).

[San+18] Rodrigo Santa Cruz, Basura Fernando, Anoop Cherian, and Stephen Gould. "Visual permutation learning". In: *IEEE transactions on pattern analysis and machine intelligence* 41.12 (2018), pp. 3100–3114 (cit. on p. 99).

[SU07] Andrew I Schein and Lyle H Ungar. "Active learning for logistic regression: an evaluation". In: *Machine Learning* 68.3 (2007), pp. 235–265 (cit. on p. 43).

[Sch+21] Lars Schmarje, Monty Santarossa, Simon-Martin Schröder, and Reinhard Koch. "A survey on semi-, self-and unsupervised learning for image classification". In: *IEEE Access* 9 (2021), pp. 82146–82168 (cit. on p. 10).

[Sch+20] Steffen Schneider, Evgenia Rusak, Luisa Eck, et al. "Improving robustness against common corruptions by covariate shift adaptation". In: *Advances in Neural Information Processing Systems* 33 (2020) (cit. on pp. 103, 121–123).

[SP97] Mike Schuster and Kuldip K Paliwal. "Bidirectional recurrent neural networks". In: *IEEE transactions on Signal Processing* 45.11 (1997), pp. 2673–2681 (cit. on p. 88).

[SSB85] Oliver G Selfridge, Richard S Sutton, and Andrew G Barto. "Training and Tracking in Robotics." In: *Ijcai*. 1985, pp. 670–672 (cit. on p. 43).

[Seo+20] Seonguk Seo, Yumin Suh, Dongwan Kim, et al. "Learning to optimize domain specific normalization for domain generalization". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*. Springer. 2020, pp. 68–83 (cit. on p. 103).

[SHK20] Hongje Seong, Junhyuk Hyun, and Euntai Kim. "Kernelized memory network for video object segmentation". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*. Springer. 2020, pp. 629–645 (cit. on p. 65).

[SFR14] Pierre Sermanet, Andrea Frome, and Esteban Real. "Attention for fine-grained categorization". In: *arXiv preprint arXiv:1412.7054* (2014) (cit. on p. 32).

[SSB15] Naveen Shankar Nagaraja, Frank R Schmidt, and Thomas Brox. "Video segmentation with just a few strokes". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 3235–3243 (cit. on p. 63).

[SK19] Connor Shorten and Taghi M Khoshgoftaar. "A survey on image data augmentation for deep learning". In: *Journal of big data* 6.1 (2019), pp. 1–48 (cit. on p. 31).

[SGG16] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. "Training region-based object detectors with online hard example mining". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 761–769 (cit. on pp. 43, 55).

[Shu+18] Rui Shu, Hung H Bui, Hirokazu Narui, and Stefano Ermon. "A dirt-t approach to unsupervised domain adaptation". In: *arXiv preprint arXiv:1802.08735* (2018) (cit. on p. 103).

[SZ14] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014) (cit. on pp. 20, 71, 81, 93, 94).

[SK67] Richard Sinkhorn and Paul Knopp. "Concerning nonnegative matrices and doubly stochastic matrices". In: *Pacific Journal of Mathematics* 21.2 (1967), pp. 343–348 (cit. on p. 107).

[Ski58] Burrhus F Skinner. "Reinforcement today." In: *American Psychologist* 13.3 (1958), p. 94 (cit. on pp. 32, 43).

[Son22] Jeany Son. "Contrastive Learning for Space-time Correspondence via Self-cycle Consistency". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 14679–14688 (cit. on p. 100).

[Soy22] Derya Soydaner. "Attention mechanism in neural networks: where it comes and where it goes". In: *Neural Computing and Applications* 34.16 (2022), pp. 13371–13385 (cit. on p. 32).

[Ste01] Carsten Steger. "Similarity measures for occlusion, clutter, and illumination invariant object recognition". In: *Pattern Recognition: 23rd DAGM Symposium Munich, Germany, September 12–14, 2001 Proceedings*. Springer. 2001, pp. 148–154 (cit. on p. 31).

[Su+19] Hang Su, Varun Jampani, Deqing Sun, et al. "Pixel-adaptive convolutional neural networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 11166–11175 (cit. on p. 31).

[Su+20] Jiahao Su, Wonmin Byeon, Jean Kossaifi, et al. "Convolutional tensor-train lstm for spatio-temporal learning". In: *arXiv preprint arXiv:2002.09131* (2020) (cit. on pp. 90, 92, 93).

[SWF+15] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. "End-to-end memory networks". In: *Advances in neural information processing systems*. 2015, pp. 2440–2448 (cit. on p. 66).

[SFS17] Baochen Sun, Jiashi Feng, and Kate Saenko. "Correlation alignment for unsupervised domain adaptation". In: *Domain Adaptation in Computer Vision Applications*. Springer, 2017, pp. 153–171 (cit. on p. 102).

[Sun+20a] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, et al. "Scalability in perception for autonomous driving: Waymo open dataset". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 2446–2454 (cit. on pp. 105, 113, 115, 116).

[Sun+20b] Peize Sun, Jinkun Cao, Yi Jiang, et al. "Transtrack: Multiple object tracking with transformer". In: *arXiv preprint arXiv:2012.15460* (2020) (cit. on p. 101).

[Sun+20c] Yu Sun, Xiaolong Wang, Zhuang Liu, et al. “Test-time training with self-supervision for generalization under distribution shifts”. In: *icml*. 2020 (cit. on pp. 103, 121–123, 129).

[Sun+14] Martin Sundermeyer, Tamer Alkhouli, Joern Wuebker, and Hermann Ney. “Translation modeling with bidirectional recurrent neural networks”. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 14–25 (cit. on p. 88).

[SVL14] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. “Sequence to Sequence Learning with Neural Networks”. In: *Advances in Neural Information Processing Systems* 27 (2014), pp. 3104–3112 (cit. on p. 2).

[SB18] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018 (cit. on pp. 26, 27, 30, 31).

[Sut+00] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. “Policy gradient methods for reinforcement learning with function approximation”. In: *Advances in neural information processing systems*. 2000, pp. 1057–1063 (cit. on p. 35).

[Sut84] Richard Stuart Sutton. “Temporal credit assignment in reinforcement learning”. In: (1984) (cit. on p. 35).

[Sze+17] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. “Inception-v4, inception-resnet and the impact of residual connections on learning.” In: *AAAI*. Vol. 4. 2017, p. 12 (cit. on pp. 20, 25, 50).

[Sze+15] Christian Szegedy, Wei Liu, Yangqing Jia, et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. doi: 10.1109/cvpr.2015.7298594. 2015, pp. 1–9 (cit. on p. 31).

[TD20] Zachary Teed and Jia Deng. “Raft: Recurrent all-pairs field transforms for optical flow”. In: *European conference on computer vision*. Springer. 2020, pp. 402–419 (cit. on pp. 89, 90).

[TC00] Dennis Tell and Stefan Carlsson. “Wide baseline point matching using affine invariants computed from intensity profiles”. In: *Computer Vision-ECCV 2000: 6th European Conference on Computer Vision Dublin, Ireland, June 26–July 1, 2000 Proceedings, Part I 6*. Springer. 2000, pp. 814–828 (cit. on p. 31).

[TH21] Atharva Tendle and Mohammad Rashedul Hasan. “A study of the generalizability of self-supervised representations”. In: *Machine Learning with Applications* 6 (2021), p. 100124 (cit. on p. 99).

[TAS17] Pavel Tokmakov, Karteek Alahari, and Cordelia Schmid. “Learning video object segmentation with visual memory”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 4481–4490 (cit. on pp. 63, 64, 88).

[Tok+22] Pavel Tokmakov, Allan Jabri, Jie Li, and Adrien Gaidon. “Object Permanence Emerges in a Random Walk along Memory”. In: *arXiv preprint arXiv:2204.01784* (2022) (cit. on pp. 101, 114).

[Tok+21] Pavel Tokmakov, Jie Li, Wolfram Burgard, and Adrien Gaidon. “Learning to track with object permanence”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 10860–10869 (cit. on pp. 101, 103, 112, 114, 115).

[Tri04] Bill Triggs. “Detecting keypoints with stable position, orientation, and scale under illumination changes”. In: *Computer Vision-ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part IV 8*. Springer. 2004, pp. 100–113 (cit. on p. 31).

[Tse+20] Hung-Yu Tseng, Hsin-Ying Lee, Jia-Bin Huang, and Ming-Hsuan Yang. “Cross-domain few-shot classification via learned feature-wise transformation”. In: *arXiv preprint arXiv:2001.08735* (2020) (cit. on p. 102).

[UVL16] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. “Instance normalization: The missing ingredient for fast stylization”. In: *arXiv preprint arXiv:1607.08022* (2016) (cit. on p. 103).

[Vas+22] Vijay Vasudevan, Benjamin Caine, Raphael Gontijo-Lopes, Sara Fridovich-Keil, and Rebecca Roelofs. “When does dough become a bagel? analyzing the remaining mistakes on imagenet”. In: *arXiv preprint arXiv:2205.04596* (2022) (cit. on p. 32).

[Vas+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017) (cit. on pp. 2, 65, 101).

[Ven+19] Carles Ventura, Miriam Bellver, Andreu Girbau, et al. “Rvos: End-to-end recurrent network for video object segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 5277–5286 (cit. on pp. 64, 73, 82, 83, 92).

[Voi+19] Paul Voigtlaender, Yuning Chai, Florian Schroff, et al. “Feelvos: Fast end-to-end embedding learning for video object segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 9481–9490 (cit. on p. 65).

[VL17] Paul Voigtlaender and Bastian Leibe. “Online adaptation of convolutional neural networks for video object segmentation”. In: *arXiv preprint arXiv:1706.09364* (2017) (cit. on pp. 63, 73, 82, 92).

[Voi+20] Paul Voigtlaender, Lishu Luo, Chun Yuan, Yong Jiang, and Bastian Leibe. “Reducing the Annotation Effort for Video Object Segmentation Datasets”. In: *arXiv preprint arXiv:2011.01142* (2020) (cit. on p. 126).

[Voi+21] Paul Voigtlaender, Lishu Luo, Chun Yuan, Yong Jiang, and Bastian Leibe. “Reducing the Annotation Effort for Video Object Segmentation Datasets”. In: *WACV*. 2021 (cit. on p. 126).

[Von+18] Carl Vondrick, Abhinav Shrivastava, Alireza Fathi, Sergio Guadarrama, and Kevin Murphy. “Tracking emerges by colorizing videos”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 391–408 (cit. on pp. 97, 99, 120, 124, 125).

[Wan+20a] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. “Tent: Fully test-time adaptation by entropy minimization”. In: *arXiv preprint arXiv:2006.10726* (2020) (cit. on pp. 103, 121–124, 129).

[Wan+21] Gaoang Wang, Renshu Gu, Zuozhu Liu, et al. “Track without appearance: Learn box and tracklet embedding with local and global motion patterns for vehicle tracking”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 9876–9886 (cit. on pp. 101, 114).

[WSH22] Gaoang Wang, Mingli Song, and Jenq-Neng Hwang. “Recent Advances in Embedding Methods for Multi-Object Tracking: A Survey”. In: *arXiv preprint arXiv:2205.10766* (2022) (cit. on p. 103).

[Wan+20b] Huiyu Wang, Yukun Zhu, Bradley Green, et al. “Axial-deeplab: Stand-alone axial-attention for panoptic segmentation”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 108–126 (cit. on pp. 18, 93, 94).

[Wan+22] Jindong Wang, Cuiling Lan, Chang Liu, et al. “Generalizing to unseen domains: A survey on domain generalization”. In: *IEEE Transactions on Knowledge and Data Engineering* (2022) (cit. on p. 98).

[Wan+23] Wenhai Wang, Jifeng Dai, Zhe Chen, et al. “Internimage: Exploring large-scale vision foundation models with deformable convolutions”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 14408–14419 (cit. on p. 2).

[Wan+18] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. “Non-local neural networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 7794–7803 (cit. on p. 86).

[WJE19] Xiaolong Wang, Allan Jabri, and Alexei A Efros. “Learning correspondence from the cycle-consistency of time”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2566–2576 (cit. on p. 100).

[Wan+19] Yiru Wang, Weihao Gan, Jie Yang, Wei Wu, and Junjie Yan. “Dynamic curriculum learning for imbalanced data classification”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 5017–5026 (cit. on p. 33).

[WKW21] Yongxin Wang, Kris Kitani, and Xinshuo Weng. “Joint object detection and multi-object tracking with graph neural networks”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 13708–13715 (cit. on p. 101).

[Wei+19] Chen Wei, Lingxi Xie, Xutong Ren, et al. “Iterative reorganization with weak spatial constraints: Solving arbitrary jigsaw puzzles for unsupervised representation learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 1910–1919 (cit. on p. 99).

[WCA18] Daphna Weinshall, Gad Cohen, and Dan Amir. “Curriculum learning by transfer learning: Theory and experiments with deep networks”. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 5238–5246 (cit. on p. 43).

[Wer74] Paul Werbos. “Beyond regression: new tools for prediction and analysis in the behavioral sciences”. PhD thesis. Harvard University, 1974 (cit. on p. 22).

[Wil+21] Benjamin Wilson, William Qi, Tanmay Agarwal, et al. “Argoverse 2: Next Generation Datasets for Self-driving Perception and Forecasting”. In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks 2021)*. 2021 (cit. on pp. 105, 113, 115, 116).

[WC20] Garrett Wilson and Diane J Cook. “A survey of unsupervised deep domain adaptation”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 11.5 (2020), pp. 1–46 (cit. on p. 102).

[WB18] Nicolai Wojke and Alex Bewley. “Deep Cosine Metric Learning for Person Re-identification”. In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2018, pp. 748–756 (cit. on p. 101).

[WBP17] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. “Simple online and realtime tracking with a deep association metric”. In: *2017 IEEE international conference on image processing (ICIP)*. IEEE. 2017, pp. 3645–3649 (cit. on p. 101).

[Wug+18] 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*. 2018, pp. 7376–7385 (cit. on pp. 64–66, 76–78, 80–83, 87, 92).

[Xia+18] Huaxin Xiao, Jiashi Feng, Guosheng Lin, Yu Liu, and Maojun Zhang. “Monet: Deep motion exploitation for video object segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 1140–1148 (cit. on p. 64).

[Xie+17] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. “Aggregated residual transformations for deep neural networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1492–1500 (cit. on p. 50).

[Xin+15] SHI Xingjian, Zhourong Chen, Hao Wang, et al. “Convolutional LSTM network: A machine learning approach for precipitation nowcasting”. In: *Advances in neural information processing systems*. 2015, pp. 802–810 (cit. on pp. 16, 69, 91, 93).

[XW21] Jiarui Xu and Xiaolong Wang. “Rethinking self-supervised correspondence learning: A video frame-level similarity perspective”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 10075–10085 (cit. on pp. 99, 100).

[Xu+18] Ning Xu, Linjie Yang, Yuchen Fan, et al. “Youtube-vos: Sequence-to-sequence video object segmentation”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 585–601 (cit. on pp. 4, 61, 63, 64, 66, 67, 71–73, 75, 79, 80, 82, 83, 86–88, 91, 92, 134).

[Xu+19] 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/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 314–323 (cit. on p. 64).

[Yan+21a] Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. "Self-supervised Video Object Segmentation by Motion Grouping". In: *ICCV*. 2021 (cit. on p. 100).

[Yan+21b] Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. "Self-supervised video object segmentation by motion grouping". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 7177–7188 (cit. on p. 100).

[Yan+18] 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*. 2018, pp. 6499–6507 (cit. on pp. 64, 73, 82, 92).

[Yan+19] Zhao Yang, Qiang Wang, Luca Bertinetto, et al. "Anchor diffusion for unsupervised video object segmentation". In: *Proceedings of the IEEE international conference on computer vision*. 2019, pp. 931–940 (cit. on pp. 65, 77, 80).

[YC99] Zhengwei Yang and Fernand S Cohen. "Image registration and object recognition using affine invariants and convex hulls". In: *IEEE Transactions on Image Processing* 8.7 (1999), pp. 934–946 (cit. on p. 31).

[YWY21] Zongxin Yang, Yunchao Wei, and Yi Yang. "Associating objects with transformers for video object segmentation". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 2491–2502 (cit. on p. 65).

[YWY20] Zongxin Yang, Yunchao Wei, and Yi Yang. "Collaborative Video Object Segmentation by Multi-Scale Foreground-Background Integration". In: *arXiv preprint arXiv:2010.06349* (2020) (cit. on p. 65).

[YY22] Zongxin Yang and Yi Yang. "Decoupling Features in Hierarchical Propagation for Video Object Segmentation". In: *arXiv preprint arXiv:2210.09782* (2022) (cit. on p. 66).

[Yao+20] Rui Yao, Guosheng Lin, Shixiong Xia, Jiaqi Zhao, and Yong Zhou. "Video object segmentation and tracking: A survey". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 11.4 (2020), pp. 1–47 (cit. on p. 61).

[Zai+22] Syed Sahil Abbas Zaidi, Mohammad Samar Ansari, Asra Aslam, et al. "A survey of modern deep learning based object detection models". In: *Digital Signal Processing* (2022), p. 103514 (cit. on p. 104).

[ZS14] Wojciech Zaremba and Ilya Sutskever. "Learning to execute". In: *arXiv preprint arXiv:1410.4615* (2014) (cit. on p. 33).

[Zen+22] Fangao Zeng, Bin Dong, Yuang Zhang, et al. "MOTR: End-to-End Multiple-Object Tracking with TRansformer". In: *European Conference on Computer Vision (ECCV)*. 2022 (cit. on p. 101).

[Zha+17] Dingwen Zhang, Le Yang, Deyu Meng, Dong Xu, and Junwei Han. "Spftn: A self-paced fine-tuning network for segmenting objects in weakly labelled videos". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 4429–4437 (cit. on p. 43).

[Zha+20a] Feihu Zhang, Xiaojuan Qi, Ruigang Yang, et al. “Domain-invariant stereo matching networks”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 420–439 (cit. on p. 103).

[Zha19] Richard Zhang. “Making convolutional networks shift-invariant again”. In: *International conference on machine learning*. PMLR. 2019, pp. 7324–7334 (cit. on p. 32).

[ZIE16] Richard Zhang, Phillip Isola, and Alexei A Efros. “Colorful image colorization”. In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*. Springer. 2016, pp. 649–666 (cit. on p. 99).

[Zha+19] Wenwei Zhang, Hui Zhou, Shuyang Sun, et al. “Robust multi-modality multi-object tracking”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 2365–2374 (cit. on p. 114).

[Zha+20b] Yizhuo Zhang, Zhirong Wu, Houwen Peng, and Stephen Lin. “A Transductive Approach for Video Object Segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 6949–6958 (cit. on pp. 64, 92).

[Zho+21] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. “Domain generalization: A survey”. In: *arXiv preprint arXiv:2103.02503* (2021) (cit. on p. 102).

[ZKK20] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. “Tracking objects as points”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 474–490 (cit. on pp. 101, 114, 115).

[Zhu+20] Xizhou Zhu, Weijie Su, Lewei Lu, et al. “Deformable detr: Deformable transformers for end-to-end object detection”. In: *arXiv preprint arXiv:2010.04159* (2020) (cit. on p. 101).

List of Figures

List of Tables

Fatemeh Azimi

Machine Learning and Computer Vision Engineer

✉ *fatemeh.r.azimi@gmail.com*
GitHub | LinkedIn

Experience

02/2023-Present **Machine Learning and Computer Vision Engineer**, *Scene Understanding, Detection and Tracking*, Torc Robotics.
- Developed camera and lidar-based 2D/3D inference pipelines, including designing, training, and evaluating object detection and tracking algorithms.
- Optimized models for real-time performance using TensorRT and deployed them on autonomous vehicle platforms, resulting in efficient models running on Orin hardware while maintaining accuracy.
- Collaborated on synthetic data generation to address corner scenarios using Diffusion models and NeRF.
- Served as Scrum Master, coordinating inter- and intra-team dependencies and facilitating planning sessions resulting in considerable impact on achieving sprint goals and product delivery.

09/2022-12/2022 **Internship**, *3D Fine-Grained Segmentation*, Adobe.
- Benchmarked 2D and 3D methods for 3D object part segmentation.
- Explored using foundation models for 3D segmentation, improving accuracy.

02/2022-08/2022 **Internship**, *Self-supervised Multiple Object Tracking*, Algolux.
- Developed methods for self-supervised tracking, achieving up to a 13% improvement over baselines.
- Led data collection for a pretraining dataset and built a pseudo-labeling pipeline, enhancing the accuracy of the tracking model.

01/2018-10/2018 **Machine Learning Researcher**, *Applications of Reinforcement Learning in Computer Vision*, German Research Center for Artificial Intelligence (DFKI).
- Designed a reinforcement learning agent that learns to focus on salient parts of images, improving image classification accuracy in cluttered visual environments.

10/2016-06/2017 **Research Assistant**, *Camera Pose Estimation Using LSTMs*, German Research Center for Artificial Intelligence (DFKI).

09/2015-09/2016 **Research Assistant**, *Invariant Coordinate Selection in Image Processing*, RWTH Aachen.

02/2015-08/2015 **Internship**, *Software-defined Networking, OpenFlow Switches and Controllers*, Ericsson.

03/2014-02/2015 **Research Assistant**, *Network Function Virtualization and Software-defined Networking*, Ericsson.

Academic Experience

04/2019-04/2022 **Teaching Assistant**, *Seminar on Applied Artificial Intelligence*, TU Kaiserslautern.

10/2021-04/2022 **Teaching Assistant**, *Lecture Series on Applications of Machine Learning and Data Science*, TU Kaiserslautern.

07/2019 **Summer School**, *Deep Learning and Reinforcement Learning Summer School*, organized by CIFAR and AMII, Canada.

09/2016 **Summer School**, *Autonomous Driving*, organized by RWTH Aachen.

09/2012-06/2013 **Teaching Assistant**, *Electromagnetism, Electronics II, Circuits and Systems I*, Ferdowsi University of Mashhad.

Education

10/2018-12/2024 **Ph.D. in Computer Science**, *RPTU*, Germany, Overall Grade: Magna Cum Laude.

10/2013-07/2017 **M.Sc. in Electrical Engineering, Information Technology, and Computer Engineering**, *RWTH Aachen University*, Germany, Overall Grade: 1.6.

09/2008-03/2013 **B.Sc. in Electrical Engineering, Telecommunications**, *Ferdowsi University of Mashhad*, Iran, Overall Grade: 17.1/20.

Ph.D. Thesis

Title **Towards Image and Video Understanding in Challenging Scenarios**.

Explored novel approaches for image and video understanding under domain shifts, occlusions, and dataset biases. Developed state-of-the-art methods for Video Object Segmentation and Multi-Object Tracking using self-supervised and deep learning techniques.

Publications

- Fatemeh Azimi, Fahim Mannan, Felix Heide, **S^{3}Track: Self-supervised Multi-Object Tracking with Soft Assignment Flow**, ECCVw, 2024.
- Fatemeh Azimi, David Dembinsky, Federico Raue, Jörn Hees, Andreas Dengel, **Q-Learning Sequential Spatial Transformer Networks for Salient Object Classification**, ICPRAM, 2023.
- Fatemeh Azimi, Sebastian Palacio, Federico Raue, Jörn Hees, Luca Bertinetto, Andreas Dengel, **Self-supervised Test-Time Adaptation on Video Data**, WACV, 2022 (oral). Also presented at SSL Workshop, NeurIPS 2021.
- Fatemeh Azimi*, Jean-Francois Nies*, Sebastian Palacio, Federico Raue, Jörn Hees, Andreas Dengel, *Spatial Transformer Networks for Curriculum Learning*, DICTA, 2022.
- Fatemeh Azimi, Federico Raue, Jörn Hees, Andreas Dengel, **Rethinking RNN-based Video Object Segmentation**, Springer Book of VISAPP, 2021.
- Fatemeh Azimi, Stanislav Frolov, Federico Raue, Jörn Hees, Andreas Dengel, **Hybrid-S2S: Video Object Segmentation with Recurrent Networks and Correspondence Matching**, VISAPP, 2021 (oral). Also presented at WiML Workshop, NeurIPS, 2020.
- Fatemeh Azimi*, Benjamin Bischke*, Sebastian Palacio, Federico Raue, Jörn Hees, Andreas Dengel, **Revisiting Sequence-to-Sequence Video Object Segmentation with Multi-Task Loss and Skip-Memory**, ICPR, 2020.
- Fatemeh Azimi, Federico Raue, Jörn Hees, Andreas Dengel, **A Reinforcement Learning Approach for Sequential Spatial Transformer Networks**, ICANN, 2019 (oral).

Computer Skills

Programming Python, C++, MATLAB

Libraries PyTorch, OpenMMLab, Detectron, Huggingface, OpenCV, NumPy, pandas, scikit-learn, SciPy

Other Tools Git, Docker, AWS, Linux, Jupyter Notebooks, LaTeX

Research Interests

2D/3D Computer Vision: Detection and Tracking, Depth Estimation, SLAM

Generative AI: Synthetic Data Generation, Diffusion Models

Augmented Reality: Neural Scene Representations

Autonomous Systems: Perception, Sensor Fusion

Honors

2014, 2016 **DAAD Degree Completion Grant**, *RWTH Aachen*, Awarded for academic excellence.

Languages

Persian Native Speaker

English Fluent

German Intermediate (B1)