Master Thesis in Computer Science

Development of advanced cloud classification and segmentation models for solar energy applications using deep learning and synthetic data augmentation

by

Raphael Gut

Carried out at:	German Aerospace Center (DLR) Institute of Solar Research Calle Doctor Carracido 44, E-04005 Almería, Spain
Supervisor:	Dr.-Ing. Bijan Nouri bijan.nouri@dlr.de
Supervising professor:	Prof. Dr. Ulrich Göhner ulrich.goehner@hs-kempten.de
Address of the author:	raphael.gut@t-online.de

Thesis submitted: 28.03.2025

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig angefertigt, nicht anderweitig zu Prüfungszwecken vorgelegt, keine anderen als die angegebenen Hilfsmittel benutzt und wörtliche sowie sinngemäße Zitate als solche gekennzeichnet habe.

Almería 28.03.2025
Ort, Datum

Unterschrift

Acknowledgments

This thesis was conducted at the Institute of Solar Research of the German Aerospace Center (Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)), within the department Qualification. The work was carried out at the DLR office in Almería, Spain

I would like to express my deepest gratitude to the people who continuously greatly supported me for my master thesis.

First and foremost, I sincerely thank Prof. Dr. Ulrich Göhner for his valueable support, and for giving me the opportunity to work on this research under his supervision. His guidance and interest in my work were instrumental in shaping this thesis.

I would also like to express my sincere gratitude for the immense support of Dr Bijan Nouri, who provided me with sublime guidance, input, support and feedback throughout the 6 months, making this time at the German Aerospace Center in Almeria such a personally and academically enriching time.

My sincere thanks also go to Yann Fabel for his continuous support and valuable input throughout my work. His technical insights and readiness to help whenever needed were greatly appreciated.

Abstract

The transition to clean and renewable energy is one of the most present and defining challenges of our time. Among all available energy sources, solar power is the most abundant, yet its availability is highly variable due to fluctuations in solar irradiance. The primary cause of intra-hour fluctuations is cloud cover, which significantly impacts local irradiance levels. To maintain grid stability, mitigate ramp events in large-scale photovoltaic sites, and optimize the efficiency of Concentrated Solar Power (CSP) plants, reliable forecasting systems are essential. Nowcasting systems address this need by providing intra-hour forecasts, enabling the anticipation of short-term variations in solar irradiance.

A common approach to Nowcasting involves using ground-based All-Sky Imagers to capture sky images, detect clouds, and track their movement to predict future solar irradiance. The accuracy of these predictions largely depends on the quality of cloud detection, which is typically performed at the pixel level. Modern deep learning-based methods have emerged as the dominant approach, outperforming traditional techniques. However, one of the key challenges of these methods is their reliance on large, high-quality ground truth datasets for training and validation. Since manually annotating such datasets is labor-intensive and time-consuming, obtaining sufficient labeled data remains a significant challenge.

This thesis aims to enhance existing semantic cloud segmentation models by incorporating temporal dependencies between consecutive image sequences captured by All-Sky Imagers. First, a semi-supervised video segmentation approach was employed to expand an existing human-annotated ground truth dataset by a factor of 20, resulting in a total of 16.170 images. Four different methods were proposed and benchmarked against each other, demonstrating the effectiveness of this approach. Additionally, a new model architecture was developed that integrates motion cues from consecutive All-Sky Imager pairs alongside the primary image, leveraging cloud dynamics and transitions to provide richer prior information for the semantic segmentation process to learn better feature representation.

Evaluation on a validation dataset confirms the efficacy of both approaches. In particular, the semantic cloud segmentation model benefits from the enlarged training dataset, achieving improvements in accuracy and Intersection over Union (IoU) by 2.6% points and 3.5% points, respectively, compared to the current state-of-the-art model. Moreover, the motion cue-enriched model significantly enhances the differentiation of cloud classes, improving detection accuracy by up to 3.3% points for previously challenging cloud types.

Zusammenfassung

Der Übergang zu sauberer und erneuerbarer Energie ist eine der größten Herausforderungen unserer Zeit. Solarenergie ist die reichlichste verfügbare Quelle, jedoch durch Schwankungen der Solarstrahlung stark variabel. Hauptursache kurzfristiger Fluktuationen ist die Wolkenbedeckung, die die lokale Strahlungsintensität erheblich beeinflusst. Um Netzstabilität zu gewährleisten, Rampenereignisse in Photovoltaikanlagen zu minimieren und die Effizienz solarthermischer Kraftwerke (CSP) zu optimieren, sind zuverlässige Vorhersagesysteme essenziell. Nowcasting-Systeme erfüllen diese Anforderung, indem sie kurzfristige Vorhersagen liefern und so Schwankungen der Solarstrahlung frühzeitig antizipieren.

Ein gängiger Nowcasting-Ansatz nutzt bodengestützte All-Sky-Imager, um Himmelsbilder zu erfassen, Wolken zu detektieren und ihre Bewegung zu verfolgen. Die Genauigkeit der Vorhersagen hängt stark von der Qualität der Wolkenerkennung ab, die meist auf Pixelebene erfolgt. Moderne Deep-Learning-Methoden übertreffen traditionelle Techniken, sind jedoch auf große, qualitativ hochwertige Trainingsdaten angewiesen. Da die manuelle Annotation solcher Datensätze sehr aufwendig ist, stellt ihre Beschaffung eine große Herausforderung dar.

Diese Arbeit verbessert bestehende semantische Wolkensegmentierungsmodelle, indem sie zeitliche Abhängigkeiten zwischen Bildsequenzen von All-Sky-Imager-Kameras berücksichtigt. Ein Semi-supervised Video-Segmentierungsansatz wurde verwendet, um einen bestehenden, manuell annotierten Ground-Truth-Datensatz um den Faktor 20 zu erweitern, wodurch insgesamt neue 16.170 Bilder-Masken Paare erzeugt wurden. Vier verschiedene Methoden wurden verglichen, um die Effektivität dieses Ansatzes zu demonstrieren. Darüber hinaus wurde eine neue Modellarchitektur entwickelt, die neben dem Primärbild auch Bewegungshinweise aus aufeinanderfolgenden Bildpaaren integriert. Dadurch werden die Dynamik und Übergänge der Wolken besser erfasst, was eine umfassendere Merkmalsdarstellung für den Segmentierungsprozess ermöglicht.

Die Auswertung auf einem Validierungsdatensatz bestätigt die Wirksamkeit beider Ansätze. Das vergrößerte Trainingsset verbessert die Genauigkeit und den IoU der Wolkensegmentierung um 2,6%- bzw. 3,5%-Punkte im Vergleich zum aktuellen State-of-the-Art-Modell. Zudem verbessert das mit Bewegungsmerkmalen angereicherte Modell die Unterscheidung der Wolkenklassen, insbesondere bei zuvor schwierigen Wolkentypen, um bis zu 3,3%-Punkte.

Contents

Acronyms

ASI All-Sky Imager. v, 5, 14, 16, 18–21, 26, 29, 33, 34, 36, 46, 48, 54

CIEMAT Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas. 16

CNN Convolutional Neural Network. 8

CSP Concentrated Solar Power. iii, 1, 16

DLR Deutsches Zentrum für Luft- und Raumfahrt e.V.. ii

DNI Direct Normal Irradiance. 1

FCN Fully Convolutional Network. 6, 7, 70

HSI hue, intensity and saturation information. 5

IoU Intersection over Union. iii, iv, 43, 44, 46–48, 51, 56, 58, 60

MAVOS Modulated Cross-Attention Video Object Segmentation. 24, 25, 28–32, 47, 51, 70, 71

MLP Multi-Layer Perceptron. 5

OSVOS One-Shot Video Object Segmentation. 10, 21, 22, 28–32, 47, 48, 51, 54, 70, 71

PSA Plataforma Solar de Almeria. 16, 17, 25, 70

PV Photovoltaic. 1

PZA Pixel-Zenith-Angle. 4

RAFT Recurrent All-Pairs Field Transforms. 14, 15, 35

RGB Red, Green, Blue. 5, 14, 35, 38, 40

SAM Segment Anything Model. 7

SOTA State-of-the-Art. 23, 42–44, 60

STCN Space-Time Correspondence Network. 22, 23, 28, 31, 32, 47, 51, 70, 71

SVM Support Vector Machine. 5

TPR True Positives Rate. 43

WMO World Meteorological Organization. 2, 3, 70

XAI Explainable AI. 61

1 Introduction

1.1 Motivation

Cloud detection and classification play a crucial role in various applications, most notably in the field of solar energy, as well as in meteorology/climatology [HRW01] and satellite downlinking operations to optical ground stations. [GKF23]

As solar energy experiences further growth in total amounts of globally generated energy, the importance of cloud detection is increasing in order to ensure a stable and controlled switch to renewable and sustainable energy production.
Solar energy is a fluctuating power source, due to its spatial and temporal variability of solar irradiance. This variability arises from two primary sources: the diurnal and seasonal changes that occur periodically, and the intra-hourly and intra-minute variations in local solar irradiance, which are predominantly influenced by clouds. [Blu+22] While the former type of variability can be relatively easily anticipated, the latter remains challenging due to the intricate dynamics of weather systems.
The two dominant technologies in solar energy production are *Photovoltaic (PV)* and *Concentrated Solar Power* (CSP). Each is affected by irradiance variability in different ways, for the former irradiance prediction is mainly of interest for ramp rate control [Wen+20] to optimise efficiency and grid stability, especially for big PV parks. [Ant+16]
On the other hand, CSP systems, which consist of large mirror structures that focus Direct Normal Irradiance (DNI) onto a receiver to heat transfer fluids, are particularly sensitive to spatial variations in irradiance. These variations can lead to unwanted temperature gradients within the complex, resulting in thermal stress on the material [STB15].

Consequently, the utilisation of cloud detection and classification plays a significant role for the development of Nowcasting systems. The purpose of these systems is to provide accurate solar irradiance forecasts with high temporal and spatial resolution for the near future, i.e. with prediction horizons up to 20 minutes in advance. It is possible to provide highly resolved predictions with a coverage of only a few square kilometres by utilising stereographic approaches and a small number of all-sky imagers mounted in close proximity to each other. [Pen+15] Coverage of larger areas, on the order of several thousand km^2, can be achieved with large-scale all-sky imager networks, as introduced by [Blu+22].

Historically, these Nowcasting systems have been built as physical models using a chain of several successive processes, which are mainly cloud detection, cloud track-

ing and ultimately cloud geolocation. [Nou+23] A more recent approach leverages generative models to produce multiple samples of potential future meteorological scenarios, enabling direct prediction of irradiance via a regression model. However, these methods are currently limited to single-camera setups, which restrict them to point predictions that only describe conditions near the camera's location. Future developments aim to integrate a network of cameras with generative models to generate spatial forecasts in the form of irradiance maps. In such systems, cloud semantic segmentation becomes a crucial step in the processing pipeline. [Fab+24]

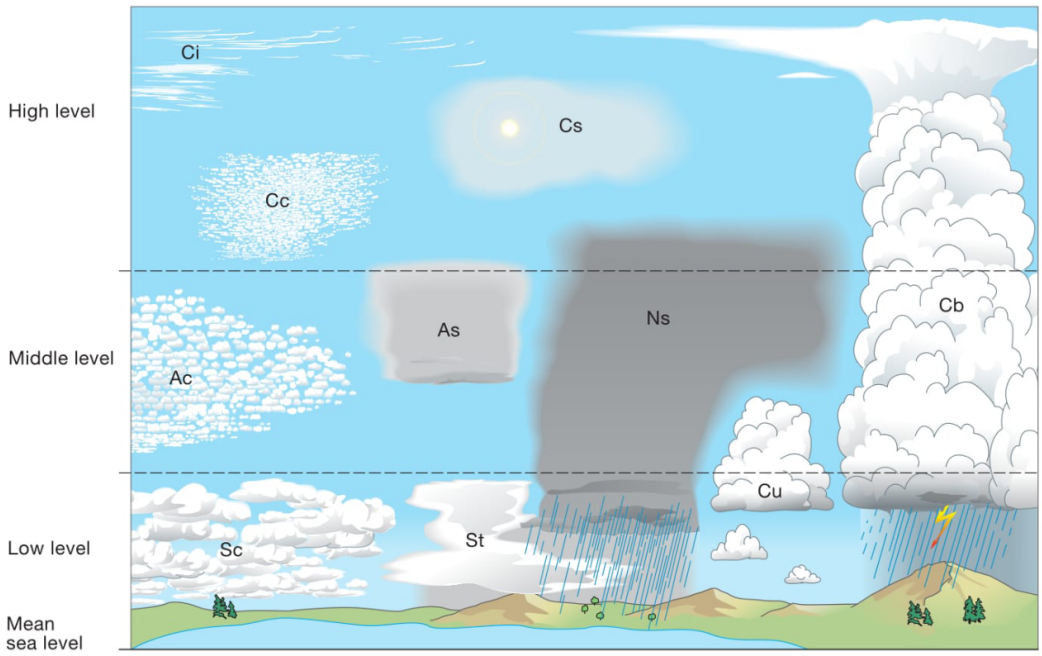
Therefore, the cloud detection step is fundamental and forms the basis for a number of subsequent steps. This requires mitigating undetected errors that would propagate through the rest of the chain and potentially up to the decision making of the target application. It is therefore of utmost importance to provide highly reliable and accurate cloud detection predictions.

1.2 Objective and challenges

This thesis aims to enhance and further develop existing methods for deep learning-based cloud detection from ground-based imagery captured by all-sky imagers. Specifically, it focuses on improving semantic cloud segmentation in these images. Semantic cloud segmentation refers to assigning each pixel in an image to one of the following atmospheric categories: *low-layer clouds*, *mid-layer clouds*, *high-layer clouds*, or *clear sky*. [Org17] By doing so the intention is not only to locate clouds within an image but to categorize pixels into groups that share visual similarities between each other.

These categories represent a significant simplification of the classification system proposed by the *World Meteorological Organization (WMO)*, which defines ten distinct cloud types, each further divided into multiple species. [Org17] A visual overview of these cloud categories is provided in figure 1, where the different cloud genera are categorized by height.

Although strongly simplified the distinguishing of clouds into one of the three categories yields considerable information about the impact of the captured cloud composition on solar irradiance. [Nou+19] High-layer clouds, such as Cirrostratus or Cirrocumulus, consist solely of ice particles and only slightly diminish the emitted solar irradiance. Whereas low-layer clouds, e.g. Cumulus or Status, are mostly dense water clouds which lead to considerable mitigating of solar irradiance. Mid-layer clouds share properties with their adjacent cloud layers, resulting in transmittance dampening between the dampening of low- and high-layer clouds.
Apart from their optical characteristics, different cloud layers often follow their own



[H]

Figure 1: Illustration of the ten main cloud genera defined by the WMO [Org17]: Cumulus (Cu), Stratus (St), Stratocumulus (Sc), Cumulonimbus (Cb), Altocumulus (Ac), Altostratus (As), Nimbostratus (Ns), Cirrus (Ci), Cirrocumulus (CC), Cirrostratus (Cs) are grouped by their base height into low-layer, mid-layer and high-layer.

specific dynamics over time, such as movement direction, velocity, enlargement, or diminution due to various wind and atmospherical conditions within the troposphere. [WRZ00] Exploiting these features to improve semantic cloud segmentation is one of the goals of this work.

Despite the extensive research that has been done on this particular task of differentiating cloud layers [Fab+22; Mag+25] , it remains a challenging topic due to several difficulties that make it hard even for human experts. Firstly, certain types of clouds share common visual characteristics and lack distinct salience, making it difficult in both single-layer and particularly complex multi-layer cloud scenarios when these layers overlap. Also distinguishing between thin high-layer clouds and atmospheric turbidity and aerosols is a challenging task due to the absence of sharp boundaries.
Another difficulty is the visual variability of the same type of clouds induced in particular by varying lighting conditions, atmospheric conditions such as atmospheric turbidity, and the fish-eye lens of the All-Sky Imager. The latter causes clouds that are close to the horizon to occupy only a small part of the image, even though they could be physically very large. This is due to the small field of view of the cameras, which leads to decreasing resolution in the image as the Pixel-Zenith-Angle (PZA) increases.
To obtain a deep learning-based semantic cloud segmentation model that performs reliably under all the potential challenges introduced, a large amount of qualitatively valuable annotated images at the pixel level representing the ground truth is required. [LSD15] Acquiring the ground truth data by manually annotating them by human resources yielded state-of-the-art results in the past [Fab+22] but comes along with the significant drawback that it is an enormously time-consuming task, making it impracticable for large volumes due to economic and time constraints. Therefore several recently carried out automatized approaches to create new ground truth data by utilizing self-supervised and weakly-supervised techniques lead to promising results. [Fab+22; Mag+25] The first part of this thesis continues the creation of automatically annotated ground truth data by exploiting temporal relationships between sequential images via semi-supervised video object segmentation.

The rest of this thesis is organized as follows: chapter 2 presents the state-of-the-art techniques for cloud detection from ground-based imagers, deep learning for semantic segmentation, video object segmentation, and optical flow. Chapter 3 examines the hardware sensors used as well as the datasets used for this thesis. Chapter 4 explains the methods developed in this thesis. Their effectiveness is discussed and evaluated in the following experimental results chapter. Finally, chapter 6 provides a conclusion and an outlook for further research in this area.

2 Related work

In this chapter, different techniques for cloud detection in ground-based imagery are presented. Then, a closer look at deep learning based semantic image segmentation and video segmentation is given. Finally, different methods for optical flow estimation and unsupervised training for deep learning based optical flow methods are analysed.

2.1 Cloud detection from ground-based imagers

Cloud classification and detection has been an active area of research since the mid-1980s, initially using satellite imagery and later, with the turn of the millennium, increasingly using ground-based imagery. [Gar88; Lon+06] Previous approaches to distinguishing cloud and sky pixels have used thresholding methods on the colour space information. A common way to measure this is to look at either the ratio or the difference between the red and blue colour channels. [Lon+06; LLY11; HMS10] This technique uses the property that the sky appears blue due to *Rayleigh scattering* and clouds appear white or greyish due to *Mie scattering*. Similar approaches also examine the green channel to extract hue, intensity and saturation information (HSI) to aid decision making. [Kaz+12; JRC15]
While these methods have been shown to be fairly effective in certain conditions, they are prone to error when the colour channels are oversaturated or the balance between the Red, Green, Blue (RGB) channels is altered.

With the rise of deep learning-based methods in various scientific and industrial fields, the first machine learning-based approaches for cloud detection were introduced by [Tar+14] and others. These studies explored the capabilities of Multi-Layer Perceptron (MLP)s and Support Vector Machine (SVM)s, demonstrating their superiority over traditional threshold-based methods. This shift laid the foundation for deep learning to become the dominant approach in cloud detection. The introduction of the *SegCloud* [Xie+20] architecture further demonstrated the benefits of using Convolutional Neural Networks for cloud detection from ground-based ASI observations by extracting high-level cloud features. Both approaches, however, only performed binary segmentation, disregarding finer differentiation of cloud genera. The first attempts at within-cloud classification were introduced by [DLW15], aiming to distinguish between thin and thick clouds, while [Fab+22] focused on classifying clouds into three layers based on the standard definition of the WMO International Cloud Atlas between low (water), middle (water and ice) and high layer (ice) clouds. [Org17]

There is still a need for further improvement in semantic cloud segmentation. There is limited research on cloud detection that incorporates both enhanced data aug-

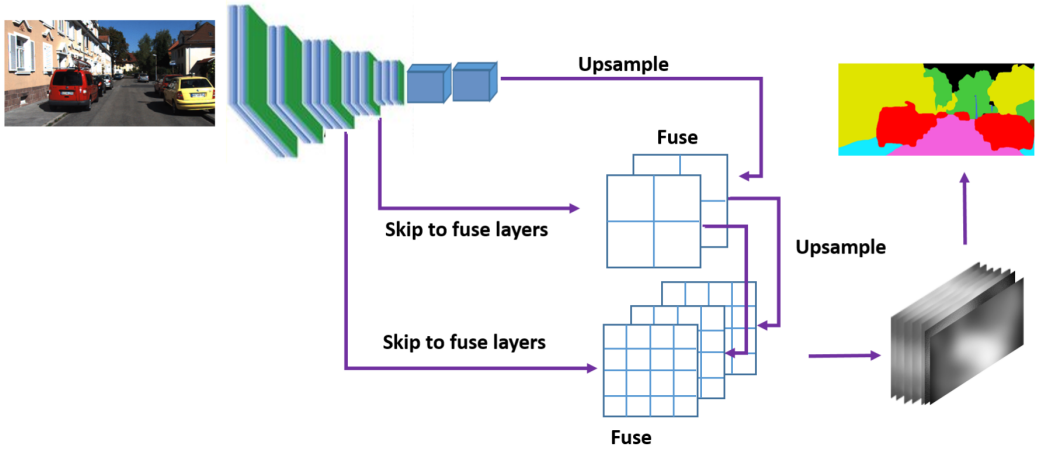


Figure 2: The FCN architecture [CVC23]

mentation and motion cues to improve segmentation accuracy. Therefore, this work represents one of the first attempts to combine semantic cloud segmentation with enhanced data augmentation and cloud motion understanding.

2.2 Deep learning for semantic segmentation

Semantic Segmentation is one of the most fundamental tasks in computer vision, that is currently undergoing major performance advances due to the highly active research in the field. It bears upon assigning each pixel in an image to exactly one class label. The task is considered inherently more challenging than image classification, where the only goal is to assign a single label to the entire image. Semantic segmentation is more than just an extension of image classification to the pixel level, as adjacent pixels are strongly correlated, thus labeling should be considered together, ultimately leading to a problem of image partitioning into semantic regions. [CVC23] The fields of application are numerous while being most prevalent in medical image analysis, autonomous driving, and robotics. [CVC23]

The foundation for modern deep learning-based semantic segmentation was laid by the *Fully Convolutional Network (FCN)* [LSD15], which transformed fully connected layers into convolutional layers, enabling the network to directly perform dense pixel-wise prediction of arbitrary input size. [LSD15]
Convolutional layers consist of convolution operations performed using filters, often in the form of small matrices (commonly 3×3, referred to as the "kernel size"). These filters slide across the entire input feature map with a defined step size, known as the "stride." Convolutional layers act as feature extractors by detecting patterns such as

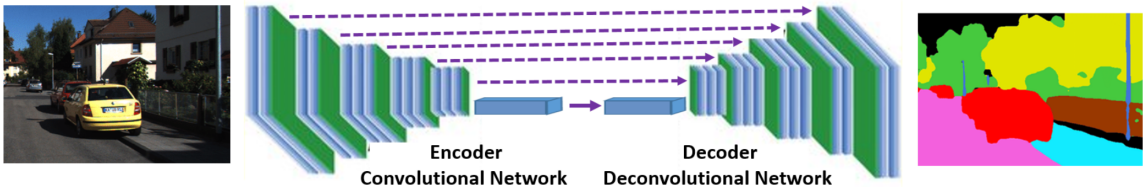


Figure 3: DeConvNet architecture [CVC23]

edges, textures, and other spatial features. [GBC16]
It has been suggested that stacking multiple convolutional layers of varying kernel sizes is beneficial for improving segmentation quality. [Eli+22] By additionally adding skip connections between the layers, the vanishing gradient problem could be efficiently addressed, allowing for the construction of very deep model architectures. [He+15] Building on the FCN approach, *encoder-decoder architectures* have emerged, where the encoder efficiently compresses the input image into a latent-space representation that captures the underlying semantic information, and the decoder generates pixel-wise predictions from this latent space. [CVC23] Skip connections between the corresponding encoder and decoder layers allow the extracted spatial information to be used by the decoder for precise localisation of the compressed features during its upsampling operations. [RFB15] The most prominent representatives are among others *U-Net* [RFB15], and *DeConvNet* [NHH15], the latter is exemplarily shown in image 3.

A widely used group of state-of-the-art architectures for highly specific semantic segmentation tasks, such as semantic cloud segmentation, is the DeepLab family established by [Che+16]. It uses a variety of techniques such as encoder-decoder structure, atrous convolution, also known as dilated convolution, and bilinear interpolation. In particular, DeepLabV3+ addresses the problem of segmenting objects at multiple scales by using multiple atrous convolutions, with different atrous rates to capture multi-scale context, called Atrous Spatial Pyramid Pooling (ASPP). ASPP enhances the ability to capture both local and global contextual information to achieve a holistically coherent understanding of the image. [CVC23; Che+16]

Recently, vision transformer-based architectures, originally proposed by [Dos+16], such as *SegVit* [Zha+22], *Swin-Unet* [Cao+21], and especially the *Segment Anything Model (SAM)* and *SAM2* models [Kir+23; Rav+24] have received considerable attention due to their strong performance. Relying on self-attention mechanisms, they aim to capture the global image context and address segmentation ambiguity at the image patch level. A major drawback of these architectures is that they require a

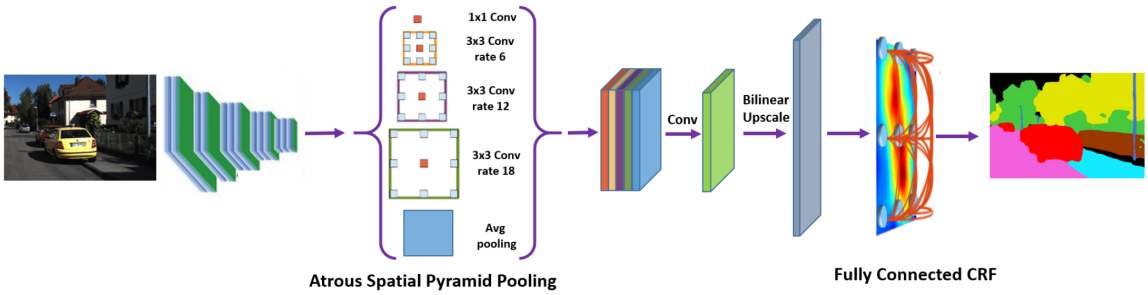


Figure 4: DeepLab architecture [CVC23]

significant amount of training data to outperform conventional Convolutional Neural Network (CNN)-based approaches, making them unattractive for semantic cloud segmentation where training data is scarce. [Ste+21]

Since deep learning based techniques learn from given training material in order to be able to generalise their acquired knowledge to unseen scenarios, large and diverse annotated data are usually required. [Roh25] Typically, this training data is obtained by human annotation, which ensures high quality but is particularly time-consuming for semantic segmentation tasks, as an annotation for each pixel in an image is required for end-to-end training, making it infeasible in industrial and research contexts when scaling to larger data quantities.

Several techniques have been developed to address this problem, using either existing annotated data or simple unannotated data, which is usually available in abundance. Traditional data augmentation approaches use the annotated training data and apply image variations such as cropping, masking, colour jittering, rotating or zooming. This allows the same data to be reused throughout the training and defies the model to actually learn to ”understand” the image rather than memorise recurring images, which would be called *overfitting*. [SK19]
To leverage unannotated data more sophisticated methods have to be employed, and therefore, in the past, studies for self-supervised and weakly-supervised approaches specifically for semantic cloud segmentation have been carried out successfully. [Fab+22; Mag+25] Another possibility is to use semi-supervised video guidance to generate new training data, which will be discussed in the following chapter.

2.3 Video Segmentation

Video segmentation refers to the identification of key objects that have some specific properties or semantics within a given video sequence. More specifically, it addresses

the problem of dividing individual video frames into multiple segments or objects. It plays an important role in many real-world applications such as autonomous driving, robotics, automated surveillance or film production. [Gao+22]

According how the output space is defined, it can be categorized broadly into two classes namely *Video Object Segmentation* and *Video Semantic Segmentation*. The former is about separating dominant foreground regions or objects (of potentially unknown category) from backgrounds while the latter, as a direct extension of image semantic segmentation to the spatial-temporal domain, aims to assign each (foreground) pixel a label within predefined semantic categories. They both share some common challenges, as occlusion, which refers to objects being obstructed by other objects in the video, deformation of objects or fast motion across the frames. [Zho+22; Kim+20]

For further separation of Video Segmentation these tasks can then be divided by indicating the level of supervision required during inference, hereby there are mainly three relevant groups [Gao+22]:

- **Unsupervised segmentation** methods perform inference without any prior knowledge
- **Semi supervised segmentation methods** initiate with the ground truth available for a few frames, most typically for the very first frame of the sequence. These labels have been annotated manually beforehand to indicate the objects to be segmented from the remaining frames.
- **Interactive methods**: In these methods, the user actively provides rough input, such as clicks, scribbles, or other annotations, to guide the model during inference.

Note that the terms *unsupervised* and *semi-supervised* are not used consistently throughout different areas in machine learning and literature, but rather have ambivalent meanings. They typically occur in the context of the actual training process of models and how much manually annotated ground truth is provided. [Zho+22; Gao+22]

This thesis employs a range of semi-supervised video segmentation approaches as a means of generating new data material. The subsequent subsection will therefore provide a more detailed examination of the underlying technique of semi-supervised video segmentation.

Semi-supervised Video object segmentation

In the early days of video segmentation, typical approaches relied among others on hand-crafted features, objectness, optical flow, and visual saliency, until the breakthrough of deep learning. [Gao+22] While these techniques were state of the art at the time, since the boom in deep-learning triggered by [KSH12], great strides have been made in terms of efficiency and accuracy, and the majority of current video segmentation models are based on deep-learning techniques. [Zho+22]
The first realization of deep-learning-based video segmentation were the *online-finetuning based methods*. This family of methods trains a segmentation model specifically for each given object mask in an online fashion, meaning that fine-tuning occurs during test time. The approach leverages the transfer learning capabilities of neural networks and typically follows a two-step procedure [Gao+22]:

- Offline training: A base model is pretrained on diverse datasets of videos and images to learn general segmentation features such as edges or textures.
- Online training: During test time, the model is fine-tuned using the annotated object in the first frame of a video sequence to learn object-specific features, enabling it to segment the object in subsequent frames accurately.

Typical representatives of this method include the *One-Shot Video Object Segmentation (OSVOS)* [Cae+21] and *LucidTracker* models [Kho+19]. However, the biggest drawback of this approach is its inability to work in real-time as online-finetuning must be performed for each new scene and object, limiting its fields of application. [Gao+22]

Therefore, shortly afterwards, more sophisticated methods were developed to address this problem, including the *optical flow based methods*. Optical flow was established as a widely used technique in VOS before the breakthrough of deep learning. This technique exploits motion patterns at the pixel level by assuming that the target objects and the background have different motion patterns. Therefore, by fusing the optical flow (e.g. by feeding the optical flow map directly into the network, as done in *MP-Net* [TAS17] or *SegFlow* [Che+17]) into a video segmentation model, it can be provided with a significant prior regarding the temporal coherence between adjacent frames, see figure 5. Although promising results were obtained at the time of publication, some problems remained, especially when the regarded object and the background flow maps are not discriminable e.g. when the object is static. [Gao+22]

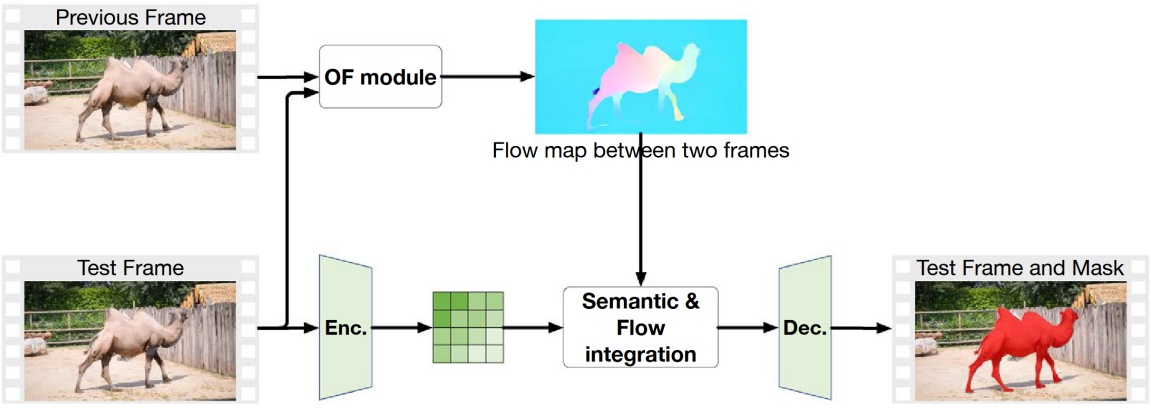


Figure 5: optical flow Semi-supervised Video segmentation [Gao+22]

Eventually, optical flow-based techniques were gradually replaced by *propagation-based methods*, where the basic concept behind this method is the use of previous frames to infer the current mask. More specifically, during inference, the segmentation model network takes the current target frame as well as the previous frame masks as input and uses them to predict object masks for the considered target frame. Subsequently, the predicted masks of previous target frames are re-propagated to facilitate the subsequent frame segmentation in a recurrent manner. (See figure 6) [Gao+22]
It is based on the assumption that target objects move spatially smoothly through the video, so that by having (location) information of the target objects from previous estimates, the model can focus more on the regions where the target object is likely to appear. As a result, the model is less distracted by background dynamics and ultimately achieves better object and background discrimination. [Gao+22]
The most famous representative of this approach is the *MaskTrack* model proposed by [Kho+16].

A different branch of semi-supervised VOS techniques are *pixel-level matching based methods*, performing inference by measuring the pixel-level correspondence between contiguous frames. [Gao+22] Two main implementation schemes for pixel-level matching have emerged, which are displayed in fig 7. Explicit matching proposed by [Che+18] computes surjectively, for each possible combination of pixel pairs, their feature similarity from the previous frame to the current frame. Thereby each pixel in the target frame is labelled with the highest similarity from all reference pixels. Therefore, the reference pixels are initialised with the values provided by the annotated first frame masks and are iteratively updated with high confidence results. Typical models with this implementation techniques are *FEELVOS* [Voi+19] and

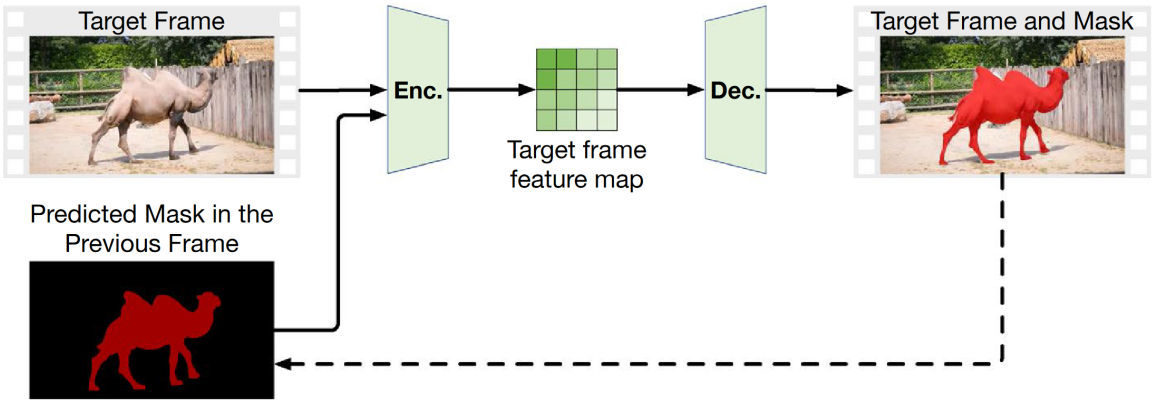


Figure 6: Mask propagation Semi-supervised Video segmentation [Gao+22]

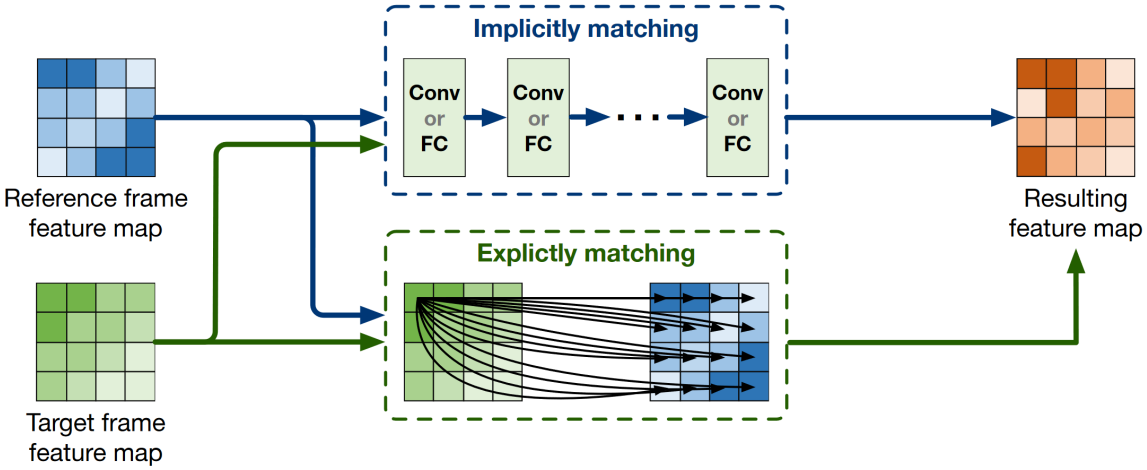


Figure 7: Pixel-level matching Semi-supervised Video segmentation [Gao+22]

VideoMatch [HHS18].
On the other hand, implicit matching, proposed by [Shi+17] designs network modules to predict cross-frame similarity holistically, rather than pixel by pixel. It uses multi-scale matching, i.e. taking features from different depth layers of the network, to extract both spatial detail and more coherent category-level semantic information between the reference and target frames to segment the target objects.

A major leap forward in semi-supervised VOS has been achieved by *STM*, which introduces a *memory-based method* that allows more past frames to be considered for pixel-level matching. [Oh+19] The main innovation here is that, unlike existing methods, previously computed segmentation information is now *stored* in an external

memory, allowing comprehensive use of the past segmentation cues. This new technique helps to overcome significant obstacles such as appearance changes, occlusions and error accumulation by learning the evolution of objects over time.

Recently, as a consequence of the development of the vision transformer [Dos+16], *transformer-based methods* have emerged. Among others, one of the first attempts to utilize vision transformer for semi-supervised VOS was *AOT* [YWY21], which also belongs to the group of pixel-level matching-based approach. By leveraging the multilayer transformer modules, multiple target objects could be encoded, matched and decoded at the same time. This mechanism supports the model to learn a global understanding, both temporarily and spatially, of the sequence, by mining and exploiting the relationships between the simultaneously existing target objects within the video sequence.

2.4 Optical flow for motion quantification in dynamic scenes

The term *optical flow* dates back to the 1950s and was originally used in psychology to describe motion perception in the visual field of animals. [Gib50] Later, with the vast innovation in the computer industry, the term was adapted in a technical way, where it established motion understanding by examining *pixel displacements* across a sequence of frames. [Alf+24]

In the early 1980s, Horn and Schnuck developed the first method for estimating optical flow, which minimises an energy function consisting of a data term and a regularisation term that ensures the smoothness of the estimated motion. [HS81] Their method is based on the *Brightness Consistency Assumption* (BCA), which states that the intensity (or brightness) of a pixel remains constant between successive frames in a sequence.
A second notable classical optical flow estimation method is that of Lucas and Kanade [LK81], which, unlike Horn-Schnucke, assumes that the *flow* is essentially constant in a local neighbourhood and can thus determine the optical flow for the pixels in that neighbourhood. Although it was developed over 40 years ago, it is still part of the famous OpenCV computer vision framework.

Following works focused on leveraging these traditional methods while also introducing modern improvements to adress challenges in complex scenarios, such as large displacements between frames.
DeepFlow for example incorporates typical deep-learning techniques, even though no training and parameter optimization is performed, such as aggregating feature information from fine-to-coarse using convolutions and max-pooling, used for matching

the correspondence to estimate motion displacement between two frames. [Wei+13]

FlowNet adopted this idea of aggregation and, as one of the first, demonstrated that it was possible to train a network to predict optical flow directly from two input images. [Dos+15] It's successor, FlowNet 2.0, was the first instance of a deep learning model surpassing the performance of classical algorithmic methods. [Ilg+17; Alf+24] Other notable networks include the Recurrent All-Pairs Field Transforms (RAFT) [TD20] model, which introduces an innovative approach using a recurrent all-pairs field transform and a recurrent method operator instead of the established spatial pyramid architectures, and FlowFormer [Hua+22], the first appearance of a transformer-based optical flow model that utilizes a self-attention mechanism to capture temporal and spatial dependencies between frames.

Unsupervised training of deep-learning based optical flow techniques
Unsupervised learning approaches for deep-learning-based optical flow estimation has become an important area of research in recent years. This is because, in contrast to other computer vision tasks such as classification, segmentation or tracking, the generation of the corresponding flow ground truth for image pairs is generally very difficult to obtain. While for rigid objects with known geometry this data can be computed, for more arbitrary scenes, such as cloud motion captured by ASIs, there is no universal approach to generating the ground truth. [Sto+21; Jon+20]

For this reason, supervised learning methods mainly rely on synthetic data for training, e.g. generated by Kubric or BlenderProc. In order to obtain high-quality synthetic training data, constructing the necessary framework environments requires careful consideration of various parameters, making it a very time-consuming endeavour. [May+18; Gre+22]
Unsupervised learning avoids the need for ground truth by using unlabelled data and optimising photometric consistency, similar to early classical methods of optical flow estimation. It has been shown that modern unsupervised learning approaches can significantly outperform classical, non-deep learning based techniques, while also being much faster at inference, since all the optimisation computation is done during training. [Jon+20]

Formally, the aim of optical flow estimation is, given a pair of RGB images $I_1, I_2 \in \mathbb{R}^{H\times W\times 3}$, to estimate the flow field $V_1 \in \mathbb{R}^{H\times W\times 2}$, indicating for each pixel I_1 the offset to its corresponding pixel in image I_2.
Where traditional methods solving an optimization problem for each image pair, unsupervised learning addresses this task by learning a function that regresses a flow

field from image pairs. More specifically the objective of unsupervised learning is to approximate a function $f_0(I_1, I_2)$ with parameters θ learned by unlabeled image sequences $D = (I_1, I_2, ..., I_n)$.
Learning the parameters θ is achieved by minimizing a loss function $\mathcal{L}(D, \theta)$, in this case by measuring the photometric consistency between an image pair (I_1, I_2) after I_2 has been warped with an estimated flow field V_1. This minimization yields the wanted parameters $\theta^* = arg\,min(\mathcal{L}(D, \theta))$.
The loss function $\mathcal{L}$ is defined as a weighted combination of multiple terms, which can be simplified as occlusion aware photometric consistency $\mathcal{L}_{photo}$ and additional regularization terms, such as edge-aware smoothness or self-supervision:

$$\mathcal{L}(D, \theta) = \omega_{photo}\mathcal{L}_{photo}(D, \theta) + \text{regularization terms} \tag{1}$$

The photometric consistency term is defined as

$$\mathcal{L}_{photo}(D, \theta) = \frac{1}{HW} \sum O_1 \odot \rho(I_1, \omega(I_2, V_1)) \tag{2}$$

, where $\frac{1}{HW} \sum$ is an abbreviated notion for the mean over all pixels, $O_1 \in \mathbb{R}^{H \times W}$ with entries $\in [0, 1]$ is the occlusion mask, which deactivates occluded pixels if they can't be reconstructed from the other image, for calculating the photometric consistency, and $\odot$ represents element-wise multiplication. The key part of the formula is the function $\rho(\cdot, \cdot)$ measuring the photometric distance between two images, one of them being warped by the function $\omega(\cdot, \cdot)$ with a flow field, which can, among others, be calculated by the generalized Charbonnier penalty function. [SRB10]

$$\rho(x) = (x^2 + \epsilon^2)^\alpha \tag{3}$$

, where $x^2 = (I_1 - \omega(I_2))^2$, while ϵ and α are constants, often initialised with $\epsilon = 0.001$ and $\alpha = 0.5$.
[SRB10; MHR18; Sto+21; Jon+20]
Popular frameworks employing unsupervised-learning techniques are UFlow [Jon+20] and Smurf [Sto+21], the latter extending the state-of-the-art optical flow estimation model *RAFT* to be trained in an unsupervised fashion.

3 Datasets of All-Sky-Imagers

In order to obtain reliable results from a data driven approach it is crucial to have a sufficiently large, high-quality and diverse training dataset. Therefore, this chapter briefly introduces the existing ground truth database and how it was created. First, the test site and hardware used for ground-based cloud observations are described.

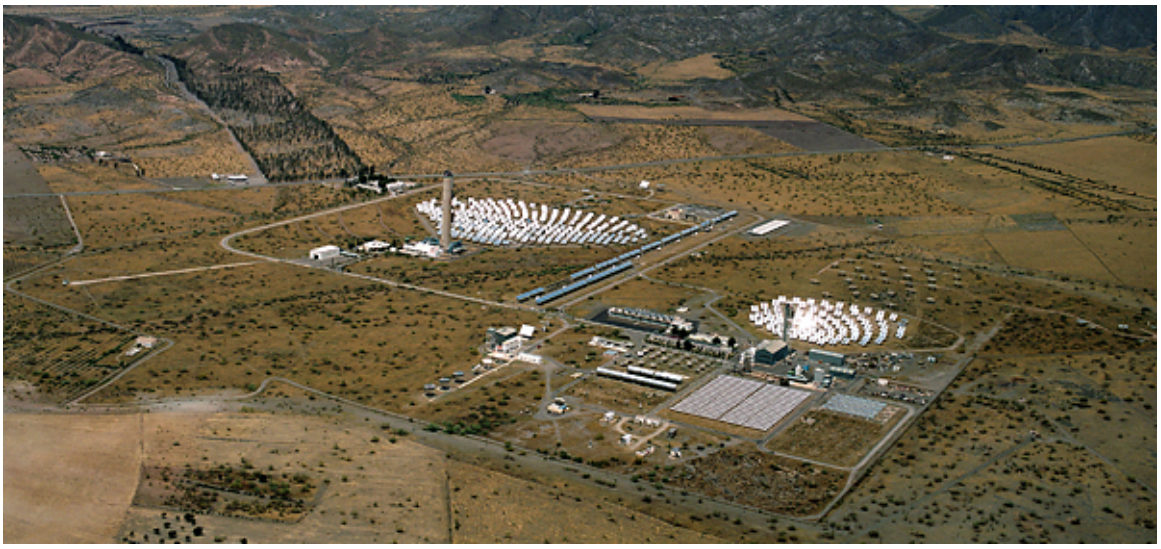


Figure 8: Aerial view of the PSA [Alm]

Then, the existing annotated ground truth data for training deep learning based cloud semantic segmentation models are presented, including the human annotated dataset with a volume of 770 masks and a dataset of about 47.000 masks generated by using weakly supervised techniques. Finally, the new dataset using Semi-supervised Video segmentation created as part of this thesis is presented.

3.1 Observation site and Hardware used

All the image data for the training data were taken with the sensor infrastructure of the Plataforma Solar de Almeria (PSA). The PSA is a solar energy test centre that is considered the largest concentrating solar power (CSP) research facility in the world. It was created in 1981 and is now run by the *Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas (CIEMAT)*. [Alm] An aerial view of the PSA is shown in 8. It is located on the edge of the Tabernas desert, surrounded by the Sierra Almahilla, the Sierra de los Filabres and the Sierra Nevada. It has the lowest aridity in Spain and Europe, making it an ideal location for solar energy. [Sol+09]
The ground-based images were captured by *All Sky Imager* (ASI) cameras, which are capable of capturing the entire visible hemisphere of the sky. The hardware used is an off-the-shelf surveillance camera from Mobotix, specifically the Q24 and Q25 models (exemplarily shown in figure 9). These ASIs are placed at various locations on the PSA. Images are taken every 30 seconds from sunrise to sunset, resulting in approximately 1.800 images per day.
In addition to the camera sensors, the creation of the weakly monitored data set re-

Figure 9: Mobotix Q25 installed on the PSA

quired the use of a ceilometer, an instrument that uses laser-based optical backscatter technology to measure cloud heights. The ceilometer emits a vertically oriented laser beam into the sky, and the device measures the time it takes for the backscattered signal to return after interacting with aerosols, cloud droplets, or other atmospheric particles. This allows the ceilometer to determine the altitude of cloud layers. Since the laser beam is focused on a narrow column, the measurement provides a vertical profile at a single location. The ceilometer model used at the PSA is a CHM15-Nimbus by Lufft, an image of it is shown in figure 10.

3.2 Human annotated and weakly annotated cloud images

The first dataset, designed for training deep learning-based semantic cloud segmentation models, comprises 770 pixel-level human-annotated, multi-label ground truth masks. The corresponding images were captured between January and November 2017.
Initially, the dataset was annotated in a binary manner, distinguishing between cloud and sky pixels, as described by [Has+20]. Subsequently, the dataset was further extended and the existing masks were refined to include the different cloud layers, specifically low-, middle-, and high-layer clouds, to enable training cloud segmentation models to differentiate between these layers. [Fab+22] The data set includes a wide range of meteorological scenarios with different cloud formations, different an-

Figure 10: Ceilometer at PSA, visible as the light blue, cuboid-shaped box next to the white structure.

gles of sun elevation and different levels of atmospheric turbidity. The corresponding data distributions are shown in Figure 3.4. The most prominent cloud layer is the low cloud layer, followed by the middle and high cloud layers. Scenarios involving two or all three layers simultaneously are far less common, which is expected to pose challenges for models attempting to generalize. Although considerable effort has been invested in human annotation, this dataset - comprising 770 images - remains relatively small for deep learning-based semantic segmentation tasks. As a reference the newly published *SA-1B* dataset, which was among other utilized to train the *Segment Anything Model* by Meta, consists of about 11 million diverse open-world images and 1.1 billion masks. [Kir+23]
To overcome the lack of more extensive and balanced training data, an additional dataset of approximately 47.000 masks was generated by [Mag+25] in a weakly labelled fashion. This is done using simple heuristics based on the ceilometer's output. If the ceilometer detects clouds in only one specific layer for an extended period, the nearby ASIs will likely detect clouds from the same layer with high certainty. This allows the cloud pixels in an ASI image to be assigned to the detected cloud layer.
This automatic labeling is feasible because binary segmentation—differentiating between sky and cloud pixels—already achieves high accuracy. The required temporal consistency of a cloud layer for the application of this heuristic was set to 4 hours

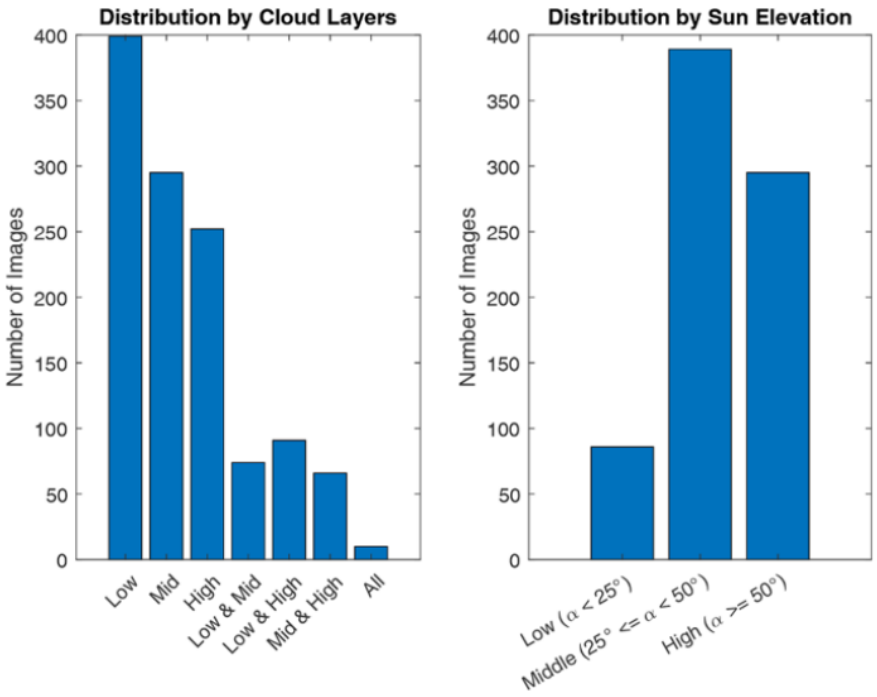


Figure 11: Data distribution of the manually annotated dataset [Fab+22]

in the past and 4 hours in the future relative to the currently regarded time point. The images for this weakly labelled dataset were taken by ASI between July 2019 and October 2021. The data distribution of this dataset is shown in Fig. 12, which shows the comparatively similar amount of low and high level clouds, while middle level clouds are under-represented. In addition, the dataset covers a wide range of atmospheric conditions, as shown by the distribution of solar elevation and turbidity.

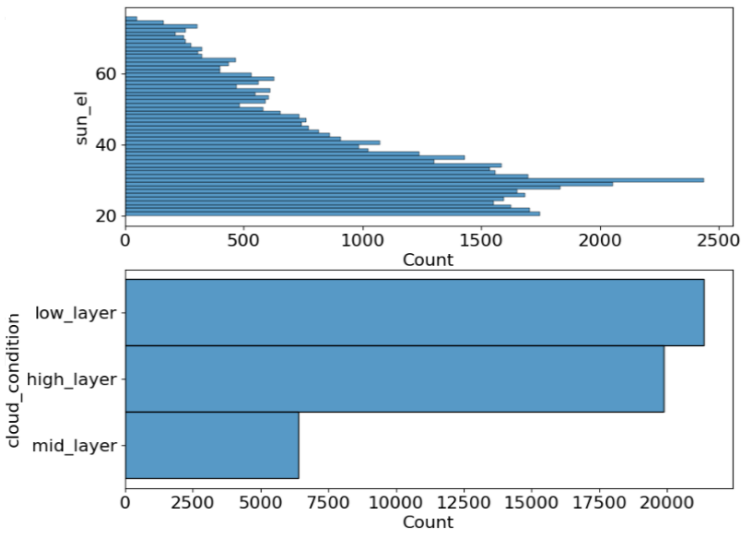


Figure 12: Data distribution of the weakly-labeled dataset [Mag+25]

3.3 Creation of automatically annotated masks

The first task in this thesis is to extend the existing ground truth dataset for cloud segmentation in ASIs by applying video segmentation techniques. While the work of [Mag+25] enabled the creation of approximately 47,000 additional image-pair masks, their impact on cloud detection did not meet expectations, despite their considerable volume. Therefore, a new technique for automatic annotation will be assessed.
The core idea is to leverage the semantic consistency and temporal continuity present in coherent ASI image sequences. Because cloud structures typically change only gradually over a short period of time, common video segmentation challenges such as fast-moving objects or sudden occlusions are reduced, resulting in more accurate predictions.
The starting point for this approach consists of 770 fully human-annotated ground-truth masks (see 3.2). These masks serve as annotated frames that act as priors for semi-supervised video segmentation, which propagates to adjacent frames. As introduced in 2.3, semi-supervised video segmentation leverages a small subset of annotated frames within a video sequence to guide the segmentation of subsequent frames.
By applying these techniques, it was possible to augment the ground-truth dataset by a factor of 20, to 15.400 newly created fully annotated images. This augmentation

was achieved by creating frame sequences centered around each of the 770 annotated frames. For each annotated frame, ten preceding and ten following frames from the associated ASI image sequence were added, resulting in sequences of twenty-one frames in total (10 preceding + 1 annotated + 10 subsequent).
The choice of using ten frames in each direction was made heuristically. It serves as a balance between the time consuming effort required for preprocessing, inference, and post-processing for every newly generated ground-truth mask, and the underlying variability of cloud dynamics over longer periods. Using significantly more than ten frames in either direction would increase computational costs and could degrade the accuracy of the propagated annotations due to the seemingly chaotic nature of cloud changes over time.

3.3.1 Introducing the Models

To evaluate the overarching approach of data augmentation via semi-supervised video segmentation, four distinct deep learning models were selected, adapted for this particular use case, fine-tuned, and eventually employed to generate the new ground-truth datasets. By incorporating multiple models, the evaluation of this approach does not depend on a single model's performance, thereby reducing the risk associated by potential, task-specific underperformance of any individual model. Furthermore, employing various video segmentation techniques (as introduced in 2.3) allows for a direct performance comparison between these four models. The selected models are:

- OSVOS
- STCN
- MAVOS
- Cutie

OSVOS
The *One-Shot Video Object Segmentation* (OSVOS) model, introduced in 2017, was the first method to enable online fine-tuning for Semi-Supervised Video Object Segmentation. [Gao+22; Cae+21] It achieved competitive results on DAVIS-2016 [Per+16] and YouTube-Objects [Xu+18], demonstrating the effectiveness of its underlying technique.
However, due to its simplicity and tendency to overfit, OSVOS is prone to insufficiently adapting to object changes and can be misled by regions that resemble the annotated frame. Despite these limitations, it serves as a baseline for comparison against the more advanced models utilized in this task. Its schematic structure is shown in figure 13.

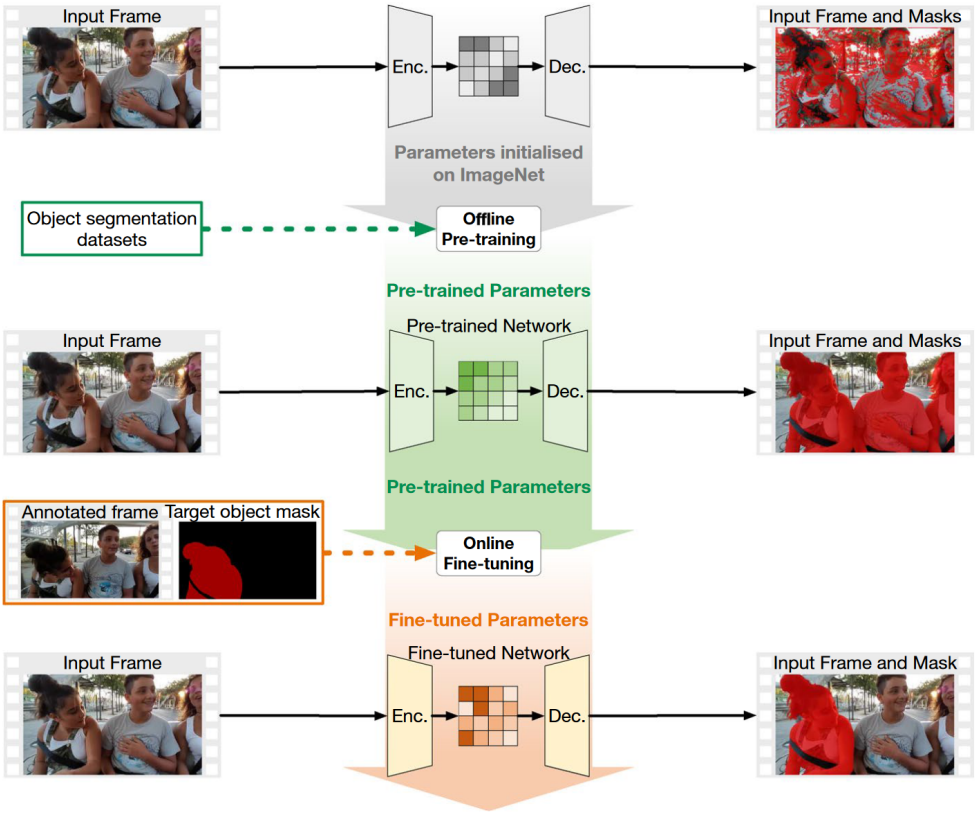


Figure 13: OSVOS structure [Gao+22]

Space-Time Correspondence Network (STCN)
The *STCN*, whose architecture is shown in figure 14, is a memory-based method and one of the most popular approaches for Semi-Supervised Video Segmentation. [CTT21] STCN performs segmentation using an attention-like mechanism:

1. It encodes key and value features from both the target and reference frames.
2. It calculates key similarities between frames.
3. These similarities are used as weights to aggregate the reference values.
4. The aggregated values are concatenated with the target features and passed into a decoder to predict the final segmentation output.

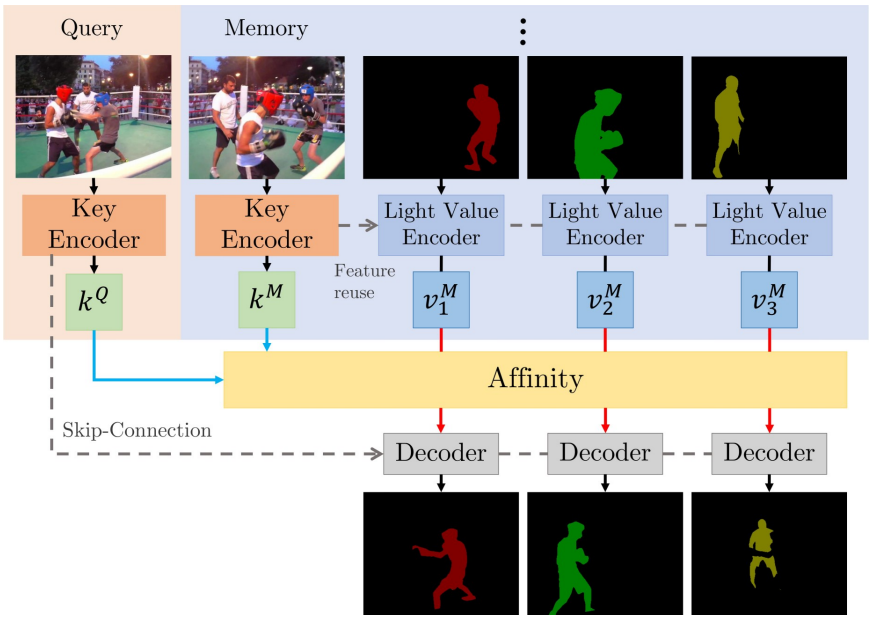


Figure 14: STCN Architecture [CTT21]

The model's effectiveness is largely attributed to its advanced encoder architecture and refined similarity measurement approach, which enable precise and efficient segmentation. [CTT21]

Cutie

Only released recently, Cutie combines memory-based and query-based approaches, leveraging pixel-level memory features alongside high-level object queries that serve as abstract summaries of target objects. [Che+24] Figure 15 illustrates the architecture. Inspired by the *XMem* [CS22] model's pixel memory mechanisms, Cutie enhances them with a novel *object-level memory reading* mechanism, which iteratively refines segmentation by integrating pixel features and object queries. This approach retains high-resolution feature maps for detailed accuracy while enabling global reasoning through object-level representations. By combining bottom-up pixel-level processing with top-down object-level abstractions in a bidirectional and iterative manner, Cutie achieves remarkable performance on challenging benchmarks such as MOSE [Din+23], setting a new State-of-the-Art (SOTA) in Video Object Segmentation. [Che+24]

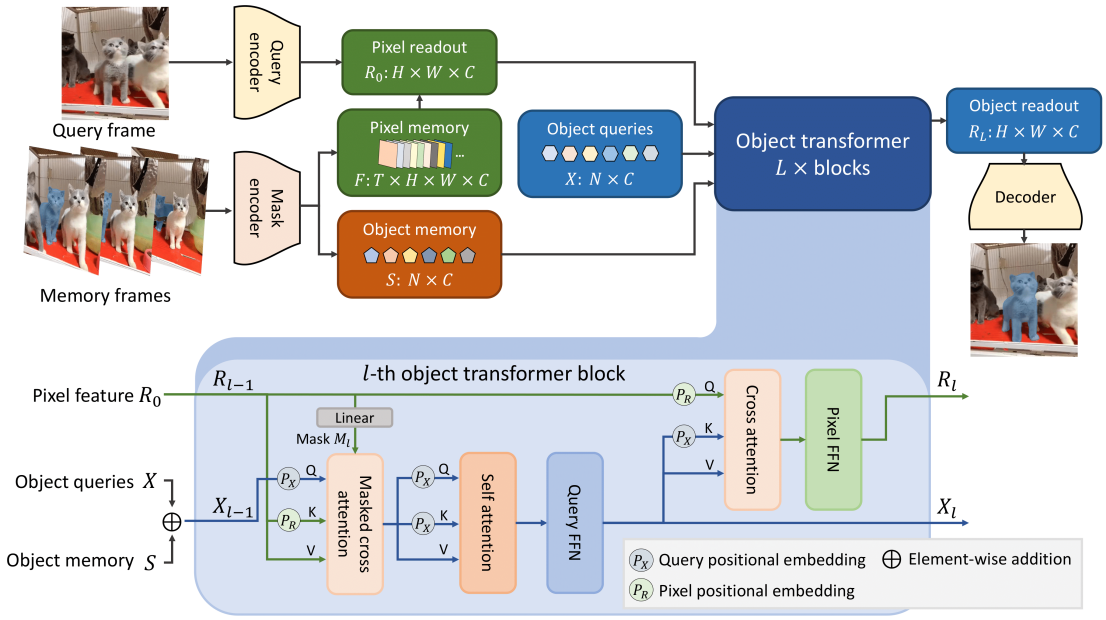
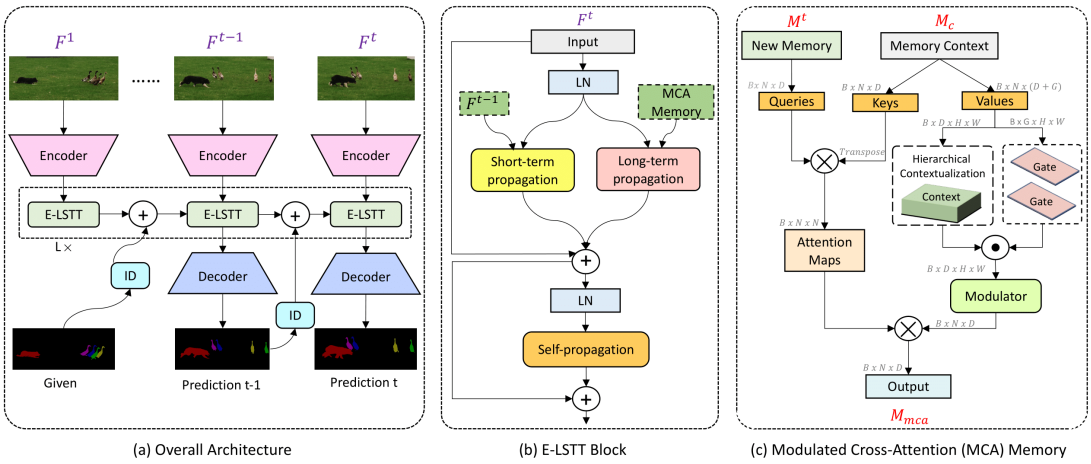


Figure 15: Cutie Architecture [Che+24]

Modulated Cross-Attention Video Object Segmentation (MAVOS)
Also released in 2024, MAVOS is a transformer-based approach designed specifically to handle longer video sequences efficiently. It introduces a novel and optimized *long-term modulated cross-attention (MCA)* mechanism, which models temporal smoothness from past frames without requiring frequent memory expansion. [Sha+24] This is achieved by retaining only relevant elements and progressively discarding irrelevant features from the long-term memory, thereby significantly reducing GPU memory consumption. The proposed MCA mechanism effectively encodes both local and global features at various levels of granularity, it then dynamically propagates only relevant information about the target while fading away irrelevant data. Additionally, MAVOS demonstrates strong real-time performance, achieving consistent speed and memory usage across both short and long video sequences, while still reaching *state-of-the-art* performance on datasets like LVOS [Hon+22] or DAVIS2017 [Pon+17]. An overview of its underlying architecture can be found in figure 16. [Sha+24]



(a) Overall Architecture (b) E-LSTT Block (c) Modulated Cross-Attention (MCA) Memory

Figure 16: MAVOS Architecture [Sha+24]

3.3.2 Mask generation procedure

The process of generating the new ground truth dataset involves several well-defined steps. These steps collectively ensure the quality and consistency of the newly introduced augmented ground truth dataset. The overall workflow can be divided into the following key tasks, which are detailed further in this subchapter:

- Data preparation
- Adapting the selected semi-supervised video segmentation models
- Fine-tuning of the different video segmentation models
- Inference of the newly trained models on the prepared raw video sequences
- Post-process and restructure the outputted masks

Data preparation
The preparation phase comprises two essential tasks that serve as prerequisites for applying deep learning techniques. These tasks ensure that the data is correctly formatted and optimized for subsequent steps in the workflow.
Preparation for the new dataset
The initial step involved converting the 770 existing ground-truth masks from their original MATLAB file format into standard image formats. During this process, static non-sky regions within the images, such as surrounding mountains or permanent structures at the PSA, were masked to ensure they do not negatively impact segmentation performance in subsequent steps. Additionally, the masks were cropped

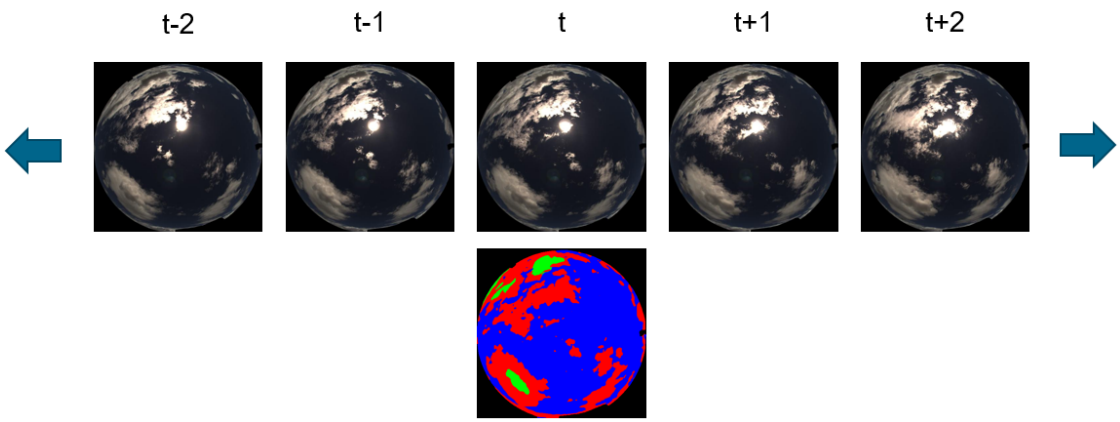


Figure 17: Sequence structuring for ground truth dataset preparation. The top row displays consecutive sky images at different time steps (from t-2 to t+2), while the bottom image shows the corresponding segmentation mask for the central frame (t).

to retain only the relevant portion of the image and finally resized.
In parallel with the mask preparation, preprocessing was also performed on the corresponding images associated with the newly generated ground-truth masks. Each ASI camera captures images at intervals of 30 seconds, forming a discrete time series. For each ground-truth mask and its corresponding image, ten frames were extracted both before and after the central frame, resulting in a 21-frame sequence spanning 10 minutes and 30 seconds. Similar to the central frame, these extracted frames were cropped to the relevant regions and resized to (512, 512). Finally the generated image sequences and corresponding ground truth masks were arranged and saved in a way that facilitates further use for training and inference. Figure 17 shows the schematic structuring of the raw image sequence along with the existing annotated mask for the centre frame.

Preparation of weakly-labeled dataset for finetuning

While initial zero-shot inference experiments with the selected models produced reasonably accurate segmentation masks, extensive fine-tuning is essential to maximise the quality of the generated output. Given that video segmentation models process image sequences, training ideally requires fully annotated segmentation sequences to provide temporal relations and contextual understanding.
Through the studies of [Mag+25] (as introduced in 3.2), an annotated, temporally coherent dataset consisting of over 47,000 images was essentially generated, based on

its heuristics of relying on unchanged cloud states over specific time periods.
Since this weakly-supervised dataset had previously been used only in a discrete manner—treating each image and its corresponding mask independently rather than sequentially—the sequences had to be restructured into self-contained, temporally coherent sequences. Previously, images were grouped based on the day they were captured, which could lead to the issue of consolidating distinct sequences from the same date, even if they represented entirely different scenes.
To address this, a heuristic was applied to ensure temporal coherence: only images where the interval between two adjacent frames was at most 5 minutes were grouped together. This approach minimized the risk of combining unrelated scenes and ensured the dataset's structural integrity for training purposes. In total, 948 distinct sequences were generated, with a median of 57 images per sequence (minimum: 5, maximum: 1,374). These sequences, along with their corresponding masks, were formatted according to the DAVIS structure, a widely used format for video segmentation tasks. This standardization enables seamless integration with video segmentation models, facilitating further experimentation and evaluation [Per+16; Pon+17].

Adapting the selected semi-supervised video segmentation models
Changes to the models were mainly focused on aligning them to ensure comparable starting conditions. This involved ensuring compatibility of the DAVIS-like structured weakly-supervised dataset with each model for the training, as well as the specifically structured image / mask pairs for inferring, which essentially required adjustments to their specific data loading modules. In addition, the data augmentation techniques and relevant hyper parameter schemes required adaptation to establish a standardized training framework across all models. These modifications ensured a consistent and fair comparison of model performance. The following table 1 detail the data augmentation techniques and hyper parameter configurations applied during the training process.

Fine-tuning of the different video segmentation models
After the necessary data and model preparations were completed, each of the four models was fine-tuned. This process was conducted on an NVIDIA RTX 2080 GPU with 8 GB of VRAM, which is considered the bare minimum for video segmentation tasks. Due to these memory constraints, the batch size was limited to 1 for all models. The fine-tuning procedure required approximately 12 to 20 hours per model.

Inference of the newly trained models on the prepared raw video sequences

Table 1: Training configuration for the different semi-supervised VOS models

	Cutie	**MAVOS**	**STCN**	**OSVOS**
Batch Size	1	1	1	1
Learning Rate	1.0×10^{-5}	1.0×10^{-6}	1.0×10^{-5}	1.0×10^{-9}
Num Iterations	150.000	150.000	150.000	150.000
Optimizer	AdamW	AdamW	Adam	SGD
Loss Function	Cross-entropy + soft dice	BCE loss + soft Jaccard loss	Bootstrapped Cross-Entropy	Cross Entropy Loss
Data Augmentation	Colour Jitter (Brightness ±10%, Contrast ±3%, Saturation ±3%)			

Once the fine-tuning phase was completed, the inference process for generating the new ground truth masks could be conducted. Several key aspects needed to be addressed. Firstly, as described, each sequence was extended temporally in **both** directions, starting from the human-annotated middle frame. This approach introduced a notable challenge: semi-supervised video segmentation models are typically designed to process single-direction sequences, meaning they only support "forward" propagation. This limitation arises because their architectures and memory mechanisms are inherently built to process sequential data progressing in a forward temporal direction. While this posed no issue for the "right" side of the sequence (subsequent frames), it created a complication for the "left" side (previous frames, see figure 17). In the left-side sequence, the starting frame with its corresponding ground truth mask effectively appeared as the last frame from the model's perspective. To address this, the sequence had to be processed in reverse order to propagate the segmentation backward to preceding frames. This challenge was resolved by temporarily reversing the order of the images in the "left" sequence. With this adjustment, the models' data loading modules could correctly process both directions of the consolidated sequence. After inference, the images were returned to their original order and structure for further usage.

Furthermore, the *OSVOS* model is specifically designed to process only binary masks and cannot handle multi-layer segmentation masks directly. In conditions with at least one cloud layer, the corresponding ground truth masks often consist of at least three distinct layers (e.g., background, clear sky, and present cloud layers). To address this limitation, the ground truth masks were split into their constituent binary layers prior to triggering the inference process. For a mask with n layers, this resulted in n separate binary masks. This preprocessing step allowed OSVOS to process each layer independently, ensuring compatibility with its technical requirements.

With these considerations in place, inference for each model was executed on the prepared data as described above. This resulted in $20 \times 770 = 15.400$ newly generated masks per model, leading to a total of $4 \times 15.400 = 61.600$ masks across all models.

post-processing and restructuring of the outputted masks
After disassembling the ground truth masks into multiple binary layers for the *OSVOS* model, the resulting inferred layers had to be reassembled into a single mask. To ensure consistency with physical and optical principles, the following logic was applied for pixel-wise reconstruction: If multiple layers predicted by the model overlap at the same pixel position, the layers were prioritized as follows:

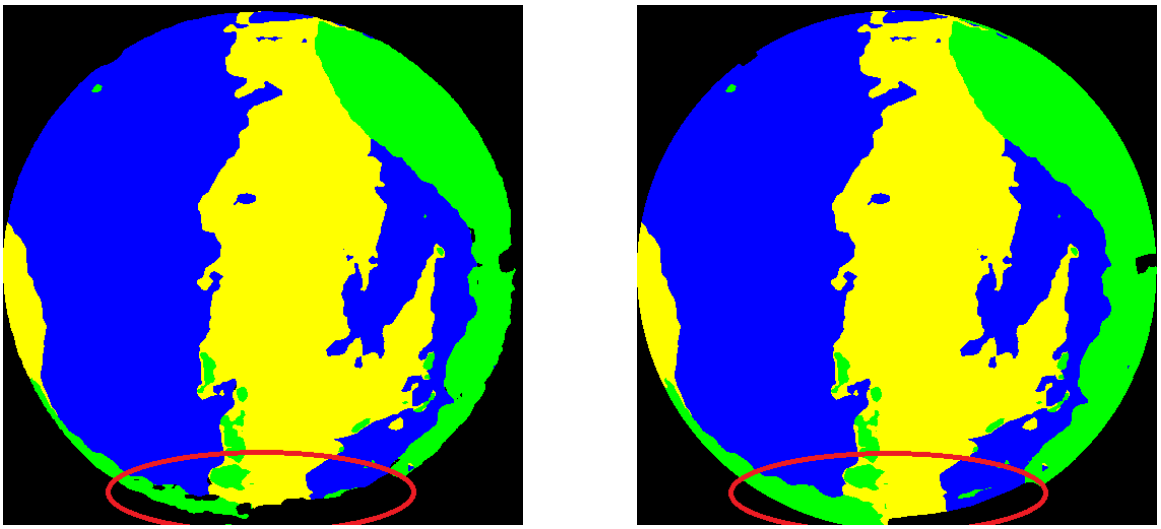
- High-layer clouds have priority over clear sky.
- Mid-layer clouds have priority over high-layer clouds.
- Low-layer clouds have priority over mid-layer clouds.

These rules are transitive from top to bottom, meaning that lower-layer clouds will always take precedence over higher layers and clear sky when combining the masks.

To further improve the quality of the generated masks, some brief post-processing steps were applied. It was observed that the masks generated by the *MAVOS* model, in particular, struggled to adhere to the basic geometries of the ASI images. To address this issue, a binary mask delineating the sky region from the border region was utilized. The following steps were applied:

1. Remove any predicted cloud/sky pixels located in the border region.
2. Enforce alignment with the sky region geometry to ensure compliance with the expected structure of ASI images.

The second step involved inpainting pixels within the sky region that were incorrectly classified as border pixels by the model. Common pre-existing inpainting algorithms, such as *INPAINT_TELEA* [Tel03] and *INPAINT_NS* [MBS02] from the OpenCV library, were tested but produced heavily noisy results that were unusable for this task. As a solution, a simple yet effective custom algorithm was developed. For each pixel requiring inpainting, the algorithm evaluated the class values of neighboring pixels within a defined surrounding region. The most frequent class value in this neighborhood was then assigned to the target pixel. This approach effectively corrected misclassified pixels while preserving the overall structure and continuity of the sky region, as can be seen in Figure 18, especially at the lower edge of the mask.



(a) raw MAVOS mask (b) postprocessed MAVOS mask

Figure 18: Effect of post-processing on MAVOS-generated segmentation masks. The raw mask (a) shows unrefined edges and misclassified regions, particularly in the red-circled area. The postprocessed mask (b) demonstrates improved boundary precision and reduced noise. Color coding: black (background), blue (sky), yellow (high-layer clouds), green (low-layer clouds)

3.3.3 Results of the mask generation and qualitative evaluation

In this subsection, a selection of generated masks is presented, discussed, and qualitatively assessed.
In figures 19 and 20 two example scenes are provided. Both display a subset of their respective frame sequences, showing the predicted masks at positions -10 min, -5 min, 0 min (ground truth mask), +5 min, and +10 min, effectively covering the full temporal span of the sequence.
The first example figure 19 shows a fairly simple weather scenario, as only one cloud layer is present. In these situations, the different models perform very similarly, at least to the human eye, and differences, even though existing, are very subtle. The quality of these predictions can be considered very good, as they capture the cloud dynamics well.
A significantly more challenging scenario arises in figure 20, where multiple cloud layers are present simultaneously and heavy cloud deformations taking place during the frame sequence. In this case, substantial differences in the predictive performance of the various models are notable, which are most distinct in the masks of *OSVOS*. Here the model struggles to separate the different cloud layers and ends up mixing

different annotations for the same cloud, resulting in a noisy prediction mask. In contrast, *STCN*, *Cutie*, and *MAVOS* deliver more comparable results throughout the sequence, even though in detail they show considerable disparities.

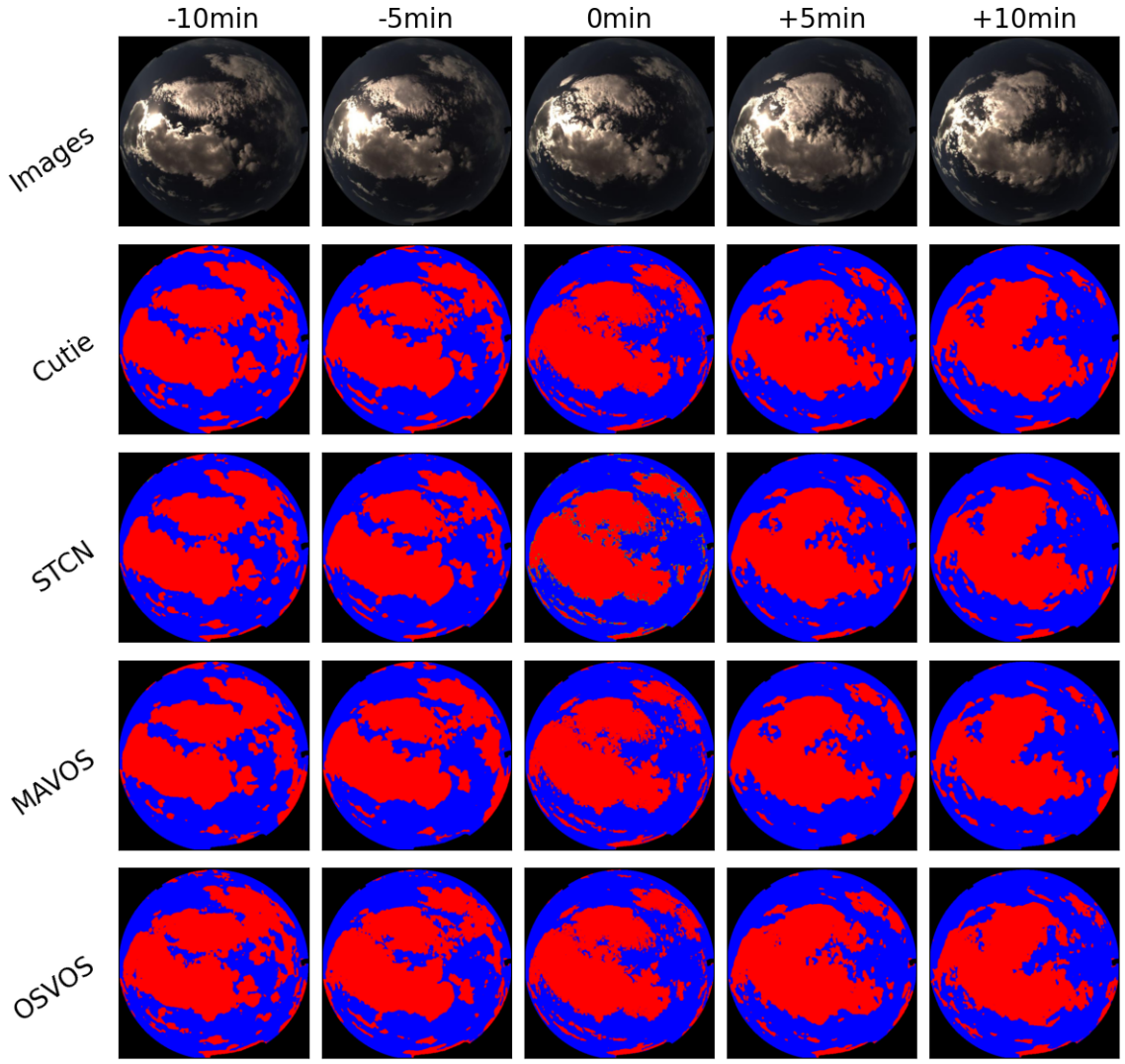


Figure 19: Visualization of automatically generated cloud segmentation masks from different models for a simple cloud condition. The top row shows input images at various time steps, while the remaining rows depict segmentation masks from Cutie, STCN, MAVOS, and OSVOS. Color coding: black (background), blue (sky), red (mid-layer clouds)

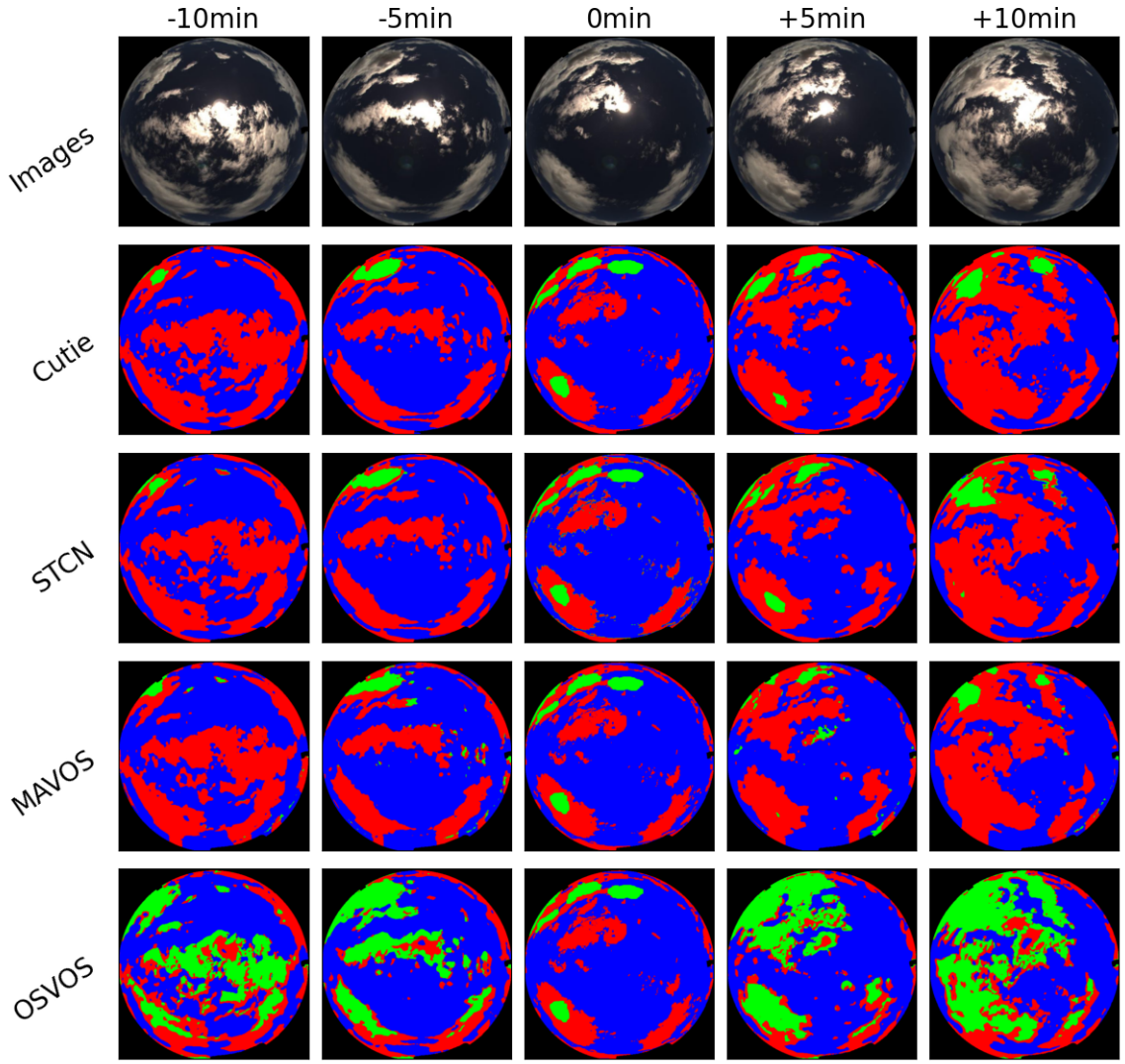


Figure 20: Visualization of automatically generated cloud segmentation masks from different models for a complex cloud condition. The top row shows input images at various time steps, while the remaining rows depict segmentation masks from Cutie, STCN, MAVOS, and OSVOS. Color coding: black (background), blue (sky), red (mid-layer clouds), green (low-layer clouds)

4 Methods

This chapter outlines the different applied methodologies and approaches for improving semantic cloud segmentation.
In the first section, the conventional method and mechanism of training a cloud segmentation model with fully supervised learning using only the human-annotated dataset with pixel-level labels is presented.

The second section of this chapter introduces a novel method for semantic cloud segmentation that incorporates motion cues from cloud dynamics extracted from ASI sequences as additional prior input during inference.

4.1 Fully supervised training of cloud segmentation models

This section describes the general procedure for training a semantic cloud segmentation model, as outlined in the chapter 2.2.
The foundation for training these models is the available training data, which provides pixel-level labels for each pixel in the images. The training procedure follows these steps: For each iteration in an epoch, a batch of images is randomly drawn from the training dataset. The size of this batch is determined by the configured batch size. For each image in the batch, the model generates predictions by assigning a class label to each pixel. In the context of this work, the possible classes are 'background,' 'clear sky,' 'low-layer,' 'mid-layer,' and 'high-layer,'.
Once predictions are made for all images in the batch, the next step is to calculate the loss, which quantifies the discrepancy between the model's predicted masks and the corresponding ground truth masks. Specifically, the *cross-entropy loss*, defined in equation 5, is used, as it is designed for pixel-level tasks. During the forward pass, the model generates pixel-wise probability maps y via a softmax function. The softmax function normalizes the raw logits $z_{n,c}$ into probabilities as follows:

$$y_{n,c} = \frac{\exp(z_{n,c})}{\sum_{c'=1}^{C} \exp(z_{n,c'})} \tag{4}$$

where $y_{n,c}$ represents the predicted probability that pixel n belongs to class C. Given these probabilities and the one-hot encoded ground truth labels $t_{n,c}$, the cross-entropy loss is calculated by

$$L_{CE}(y,t) = -\sum_{n=1}^{N}\sum_{c=1}^{C} t_{n,c} log(y_{n,c}) \tag{5}$$

The cross-entropy loss is then calculated by taking the negative logarithm of these predicted probability for the target class at each pixel n. $t_{n,c}$ is a one-hot vector

element, where $t_{n,c} = 1$ if the current c matches the ground truth label of the pixel, otherwise $t_{n,c} = 0$. Through this only the predicted probability of the target class affects the calculated loss. The cross entropy-loss approaches the value 0 the closer the predicted probability reaches 1. [Jad20]

Once the cross-entropy loss is computed for each image in the batch, the model parameters are updated to minimize this loss. By adjusting the parameters in this way, the model's predictions are guided to better align with the ground truth.
These updates are performed using gradient descent and backpropagation, implemented through the *AdamW* optimizer. AdamW builds on the Adam algorithm—an adaptive gradient method that combines elements of stochastic gradient descent (with momentum) and RMSProp to scale learning rates for individual parameters. Specifically, Adam computes a running average of gradients (first moment) and a running average of squared gradients (second moment). The AdamW variant improves upon this by decoupling weight decay from the gradient-based updates, which often leads to better generalization in practice. [LH17]

When each sample has been drawn from the training data set, an epoch is complete. After an epoch is completed, several metrics are computed on a separate validation data set to track the training progress. In particular, the validation loss is of importance; if it starts to increase for several consecutive epochs, training is stopped, since further minimisation of the training loss of the training data will lead to overfitting.

4.2 Development of motion-cued cloud segmentation models

Exploiting the distinct motion patterns of the three different cloud layers to obtain motion cues for cloud segmentation is the second major task of this thesis. As introduced in chapter 2.4, the direction and magnitude of moving objects within an image sequence can be quantified using optical flow methods. Building on these insights, this chapter investigates how to integrate such motion cues into cloud segmentation models to provide them with additional prior information—potentially improving upon purely static segmentation approaches.

4.2.1 Generating and processing optical flow maps of ASI sequences

This subsection outlines the steps taken to acquire optical flow estimations from image sequences using a state-of-the-art deep-learning-based approach. The necessary preprocessing procedures are described first, followed by the details of the training process. Subsequently, a selection of exemplary generated optical flow maps is shown. Finally, the post-processing of these raw optical flow maps is discussed in detail.

Choosing the optical flow method
A multitude of approaches exists for optical flow estimation, exacerbating the selection of a suitable method. For the purposes of this work, the modern *RAFT* model [TD20] is employed, which is provided in an *unsupervised* training framework within *Smurf*. [Sto+21]

Preparation of the training dataset
The training process for the Smurf model relies on the image sequences introduced in chapter 3.3, comprising a total of 16.170 images, 15.400 corresponding image pairs respectively.
Because the training is unsupervised and thus requires no ground truth, it would be conceivable to use all consecutive image pairs that were captured throughout the years, rather than limiting it on this particular dataset. While this approach would likely yield superior results, the required hardware resources for data preprocessing and model training would vastly exceed those available for this thesis. Consequently, the studies were initially conducted using only the 16,170 images. Moreover, the necessary preprocessing steps—such as cropping, masking, and resizing—had already been performed for the studies described in chapter 3.3.

Unsupervised training of the optical flow model
Finetuning of the Smurf model was conducted on 2 RTX 3090 with 24GB VRAM each, the used hyperparameters are listed in table 2. In total, the training of 15.000

Table 2: Training configuration for finetuning the Smurf model

Hyperparameter	**Value**
Batch Size	4
Learning Rate	2.0×10^{-4}
Num Iterations	15.000
Optimizer	Adam
Dropout Rate	0.25

steps took about 40 hours. In the following figures 22 a sample of resulting optical flow maps created by the finetuned RAFT model are shown. Because these flow maps inherently have two channels (horizontal and vertical displacement), but three channels are required for visualization in RGB, the rendering technique from [Bak+11] is used. In this representation, the flow's direction is encoded by hue, mapping opposite directions to complementary colors, while saturation and brightness correspond to the magnitude of the flow vectors (see figure 21).

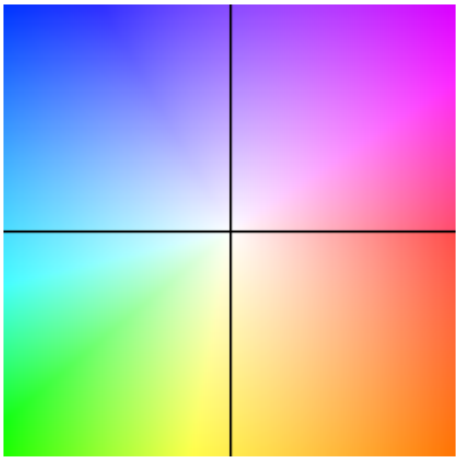


Figure 21: Optical flow color wheel. Motion direction is encoded by hue, with opposing directions mapped to complementary colors. Saturation and brightness represent motion magnitude, where more intense colors indicate higher speeds.

Post-processing of the flow maps

The figures in figure 22 show that the estimated flow lacks sharp boundaries of moving clouds, instead the captured objects in the flow maps are blurred between sky and clouds. In addition, large parts of the flow map that actually belong to sky pixels in the corresponding image are sometimes assigned flow values because the model struggles to ”understand” that the sky is fixed and confuses the relative motion of the clouds with the alleged motion of the sky.

In order to prevent the final segmentation model from misidentifying the sky as clouds due to seemingly underlying motion, the generated optical flow maps are post-processed to address these shortcomings. The post-processing encompasses the following steps:

- predict the binary mask, distinguishing cloud and sky, of the corresponding ASI image
- set the motion of the detected sky pixels in the binary mask to zero
- convert the optical flow map into a grey scale image
- compute the histogram over the frequency of the grey scale integer values and define bins
- aggregate pixel values of each bin in the fitted optical flow map to reduce complexity

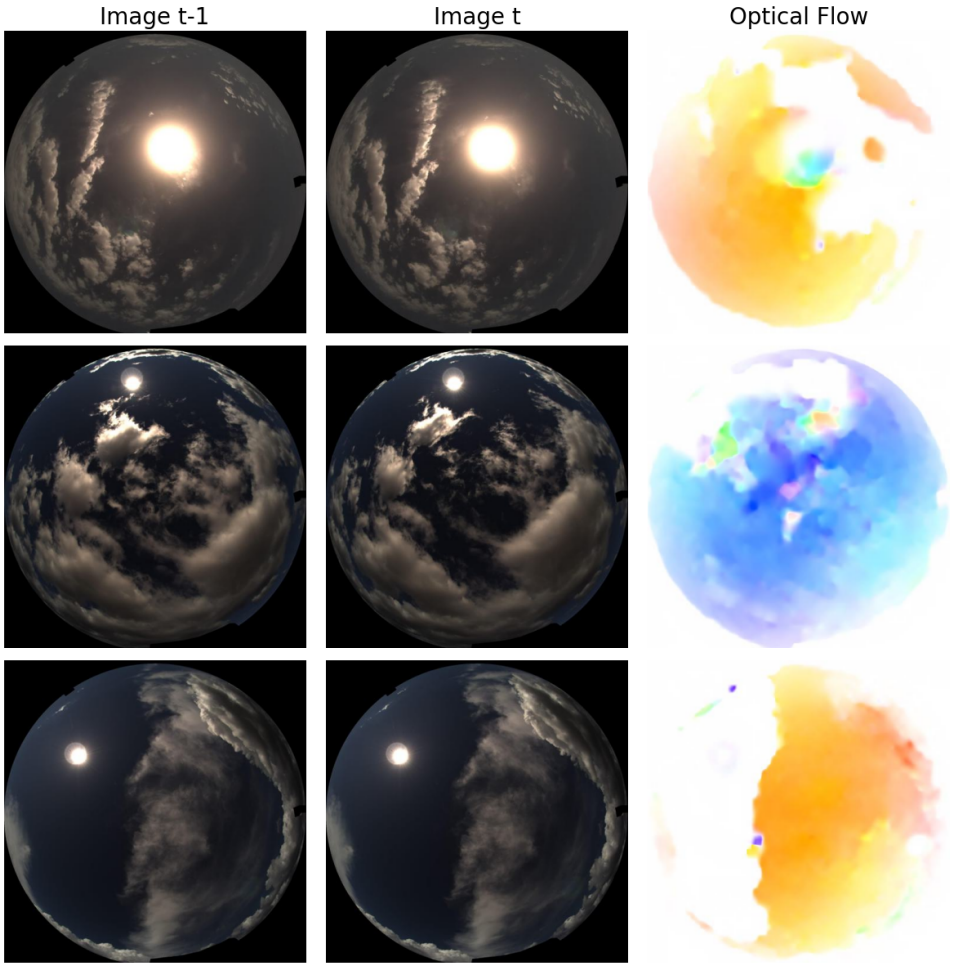


Figure 22: Optical flow samples: column 1 image t, column 2 image $t+1$, column 3 resulting estimated optical flow map

The binary mask is predicted by using the best model of the chapter 5.3, i.e. the cloud segmentation model with the augmented dataset of Cutie with factor 6. Since the model outputs multiclass labels, to achieve binary segmentation, all cloud layers are simply combined as *clouds*. Specifically, background and sky pixel are assigned the value 0 and each cloud pixel is assigned the value 1. Doing so proves to be handy for the following step to purge motion values from the sky pixel in the flow maps. This can be achieved by simply element-wise multiplying this binary mask $\in [0, 1]$ with the flow maps, which is possible due to their coinciding resolution. Since 1 is the identity element in $\mathbb{R}$, each element in the flow maps remains the same if the corresponding value in the binary mask is 1, otherwise the value becomes 0.

By converting the coloured optical flow map to greyscale, the directional information is lost; however, the intensity of motion remains, represented by the brightness of the gray tones. By examining the distribution of these greyscale intensities - in the range of $[0, 255]$ - it is possible to consolidate multiple pixel values. For that different bins can be defined by identifying the n most prominent local maxima in the distribution. The bins are then defined by the minima between these maxima. Finally, for each bin in the greyscale image, the corresponding pixel locations are used to index the original RGB optical flow map, and the mean RGB value is calculated for those pixels. This vastly reduces the complexity of the optical flow map, while retaining the essential motion information. In figure 23, the final resulting flow map after post-processing is compared to the original flow map.

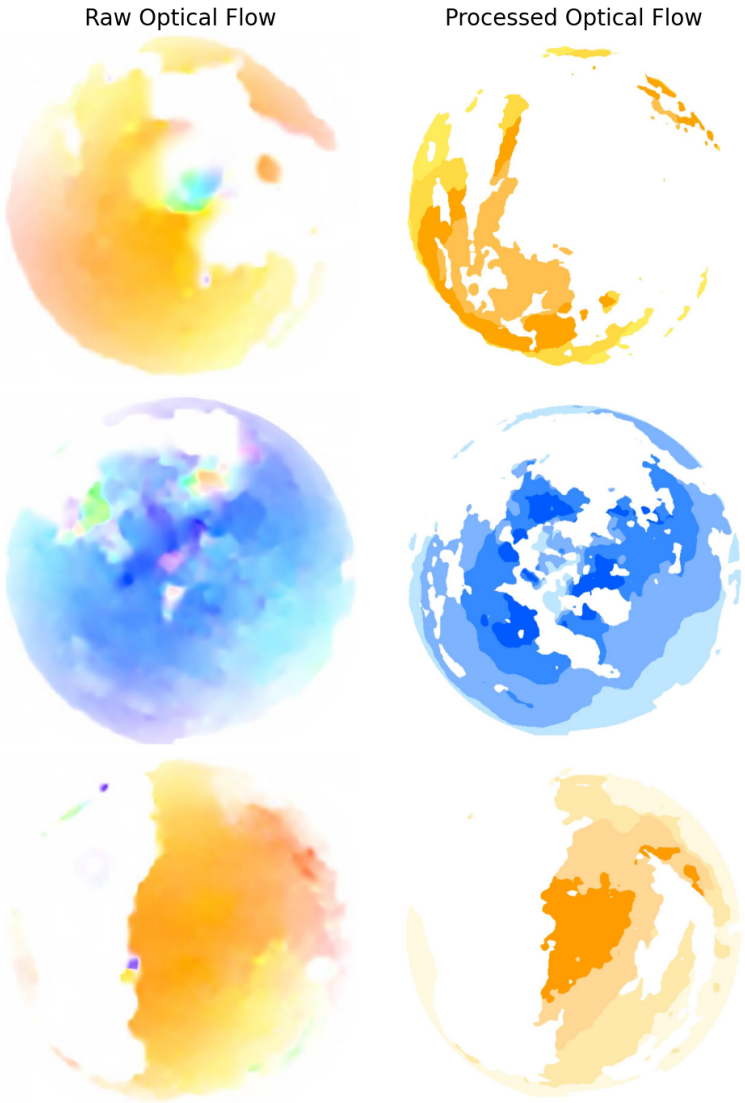


Figure 23: Optical flow estimation raw (left column) vs processed (right column)

4.2.2 Implementation of the models

The simplest approach to merge the spatial (RGB image) and motion (optical flow maps) information is to stack the RGB tensor of the input image I_t with the calculated optical flow field V_t, as can be seen in figure 24, resulting in a tensor of dimension $\mathbb{R}^{H \times W \times 5}$ (3 channels of RGB + 2 channels of optical flow maps). This consolidated tensor can now be fed directly into a machine learning model, with only minor modifications to existing implementations of the DeepLabV3-based cloud segmentation models. Specifically, the dimension of the input channels needs to be increased from 3 (RGB only) to 5 (RGB + optical flow). Modern deep-learning frameworks such as *PyTorch* can then automatically adapt the model architecture based on these changes. While this approach is very simple to implement, it has been critically discussed in several research sources. It is argued that the first layers of feature extractors attempt to capture high level spatial features. By adding low-level motion information to the input channels, the feature extraction may discard too much motion information, or even be confounded by it. [SZ14]

For this reasons a *two-stream model* as a second method is implemented and evaluated. In figure 25 the basic scheme of the two-stream method is exhibited. The core idea is to have two separate encoders for the spatial and the motion channel respectively, which gets merged together before they are fed in a decoder with classification head on top. [SZ14] The encoders usually resemble each other [FPZ16], they must have at least the same plane output dimensions to be merged pixel by pixel. Various methods have been proposed for merging the two streams, especially when dealing with different depth dimensions, which is the case for RGB and Optical Flow inputs, with *Conv-fusion* using 1×1 filters proving to be the most efficient approach compared to e.g. *Sum-fusion* or *Max-fusion*. [FPZ16]

The implemented architecture, displayed in figure 26, is divided into two sections. First, the optical flow maps are generated using the pretrained *Smurf* model, which creates the corresponding optical flow estimations for every image pair in the input stack. Second, the semantic segmentation part is implemented using the DeepLabV3 architecture. Two instances of the encoder are created for the spatial and motion branches, respectively. The outputs from both branches are merged at the end of the encoders using Conv-fusion before being fed into the DeepLabV3 decoder. This decoder then creates the pixel-wise predictions.

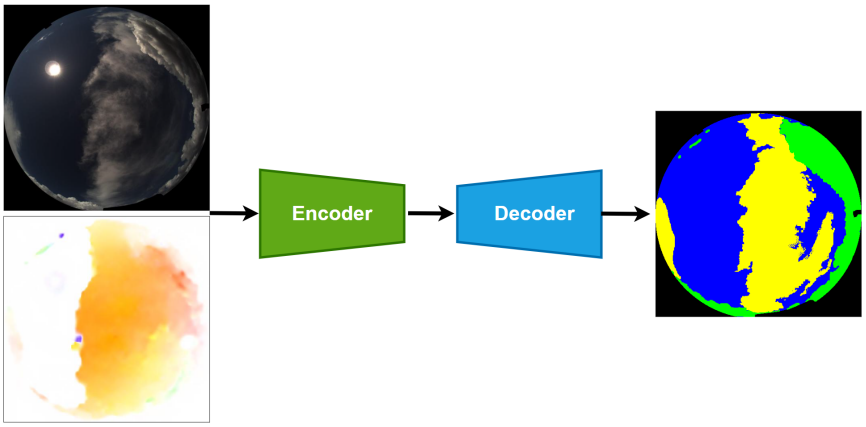


Figure 24: High-level scheme for the stacked approach to combine spatial and motion information for semantic image segmentation.

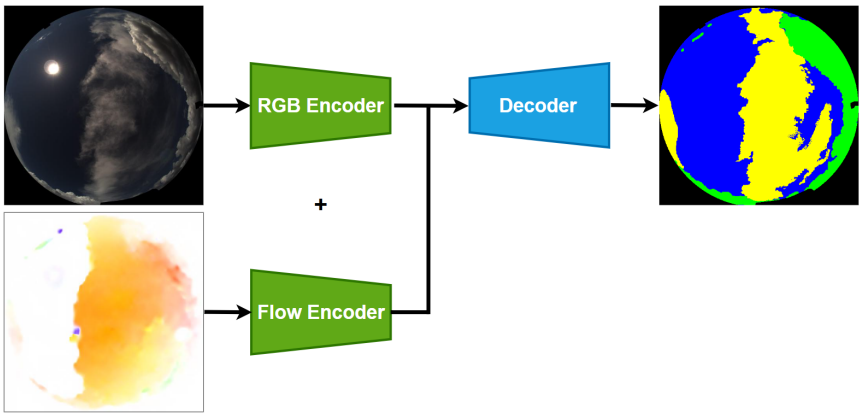


Figure 25: High-level scheme for the two-stream approach to combine spatial and motion information for semantic image segmentation

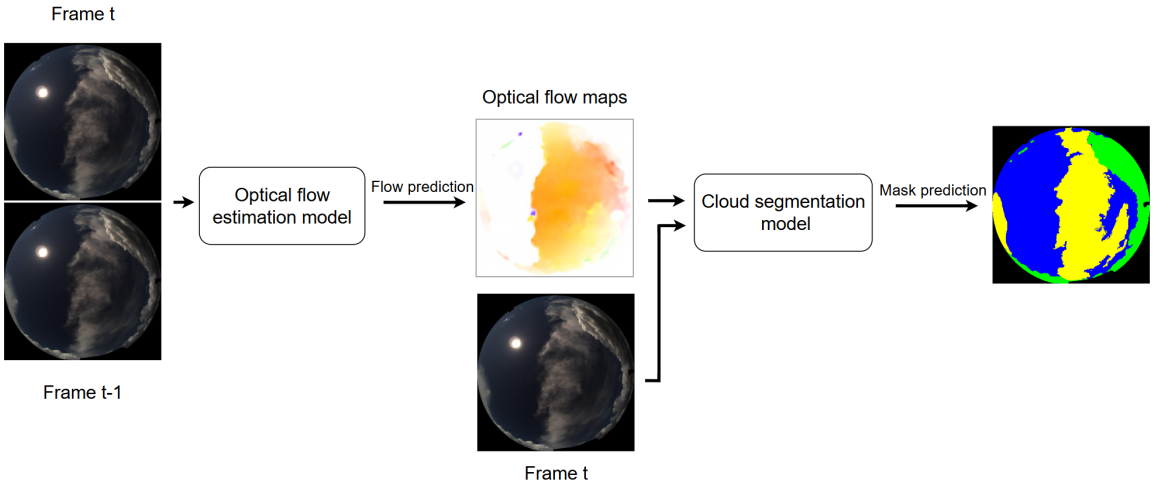


Figure 26: High-level overview of the overall process for motion cue enriched cloud segmentation

4.2.3 Training of the models

The training procedure differs from the one presented in 4.1 by incorporating multiple frames as input. However, the pixel-wise loss is computed only for the top-most frame in the stack.
In both the naive stacked approach and the two-stream approach, the optical flow estimation component is not trained, as it has already been extensively pre-trained. Despite this, both approaches—built upon the DeepLabV3 architecture—support end-to-end training. This means the models are trained as part of a unified process, where the optimization adjusts all system parameters to minimize the overall pixel-wise loss.

5 Experimental Results

Accurate semantic segmentation is a critical component of improving short-term weather predictions (Nowcasting), as it directly influences the accuracy of subsequent forecasting steps. The cloud segmentation step is used in several steps in the traditional physical model. It is therefore necessary to provide highly accurate segmentation results to improve the overall quality of the Nowcasts.
The main objective of this thesis is to tackle this task of improving the performance of the current SOTA in terms of its predictive performance for semantic cloud segmentation. Therefore, two approaches have been used in this thesis to address this issue. First, the currently existing ground truth dataset for training and evaluation of the

existing SOTA has been significantly augmented by utilizing semi-supervised video object segmentation techniques. In addition, a novel temporal motion-cue approach was established that not only considers a single frame during inference, but also takes into account additional preceding frames to capture the temporal evolution of cloud patterns and dynamics.
This chapter evaluates the effectiveness of both approaches compared to the existing SOTA model.

5.1 Utilized Software

All semantic cloud segmentation models developed and used in this thesis were implemented using the deep learning framework *PyTorch Lightning*, which provides a high-level interface for PyTorch. PyTorch Lightning is a lightweight framework created to enable training scalable deep learning models that can easily run on distributed hardware while remaining user-oriented. [FT19]
PyTorch is a popular machine learning library for the Python programming language, used for a variety of deep learning applications, such as natural language processing or computer vision, developed mainly by Facebook's AI Research Lab. [Pas+19]
The metrics used were imported by the implementations from torchmetrics, a library that provides standardised metrics for evaluating deep learning models. [Nic+22]

5.2 Definition of Common Metrics

In this subsection, the most commonly used metrics for image segmentation are introduced, namely *Accuracy*, *Precision*, *Recall*, and *Intersection over Union (IoU)*.
Accuracy refers to the ratio of correctly predicted pixels to the total number of pixels, whether positive or negative. In other words, *Accuracy* determines how often the model's predictions are correct overall. It is defined as:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{6}$$

where TP denotes true positives, TN true negatives, FP false positives, and FN false negatives.
The *Precision* measures the proportion of predicted positives that are correctly identified. For segmentation, this means the proportion of pixels classified as a specific class that actually belong to that class in the ground truth. It is defined as:

$$\text{Precision} = \frac{TP}{TP + FP}. \tag{7}$$

The *Recall*, also known as the *True Positives Rate (TPR)*, measures the proportion

of actual positives that are correctly identified. For segmentation it means the proportion of pixels in a class in the ground truth that were classified as this class. It is defined as:

$$\text{Recall} = \frac{TP}{TP + FN}. \tag{8}$$

In the context of image segmentation, these metrics can be calculated for the entire image to provide an overall evaluation, or separately for each semantic class to assess the model's performance for individual classes.
The *Intersection over Union (IoU)*, also referred to as the *Jaccard Index*, is a widely used metric for evaluating the similarity between two sets. In the context of image segmentation, it quantifies the overlap between the predicted segmentation and the ground truth. It is defined as:

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} = \frac{TP}{TP + FP + FN}, \tag{9}$$

where the *Area of Overlap* represents the intersection of the predicted and ground truth regions, and the *Area of Union* represents their combined area.
The average IoU across all classes is calculated by taking the weighted average of the class-wise IoU values:

$$IoU_{avg} = \frac{\sum_{j=1}^{C} IoU_j \cdot w_j}{\sum_{j=1}^{C} w_j} \tag{10}$$

, where C denotes the number of classes and $w_j = TP_j + FN_j$ the weighted term.

Finally, the concept of the confusion matrix is explained. Confusion matrices are a helpful tool for visualizing the strengths and weaknesses of models for semantic segmentation. They are structured as square matrices in which the rows refer to the ground truth classes, and the columns represent the classes predicted by the model. The diagonal elements represent the true positive values. The false positives for each class are found in the respective column, excluding the diagonal element, while the false negatives for each class are all elements in the respective row, excluding the diagonal element.
Intuitively the *recall* for each class can be determined from the confusion matrix by calculating $\frac{\text{diagonal element}}{\text{sum of the row}}$ and the *precision* by $\frac{\text{diagonal element}}{\text{sum of the column}}$.

5.3 Evaluation of the data augmentation via automatic annotations

The first major task of this thesis is to enhance state-of-the-art (SOTA) cloud segmentation models by augmenting their training data with automatically created data, as

described in chapter 3.3. For each of the four selected Semi-supervised Video Object Segmentation models, the dataset was expanded from 770 existing images to a total of 16,170 images. The following section provides an in-depth analysis of this approach, focusing on comparing the performance of the different models and determining the optimal augmentation factor.

Training routine

In order to obtain a fine-grained analysis of the effect of adding more ground truth data to the training pool, a number of different models were trained. For each of the four datasets generated by the different models, introduced in 3.3, ten different fully supervised semantic segmentation models were trained by gradually increasing the augmentation factor by one. The training datasets consisted of the original 770 ground truth images + $2 \cdot$the current factor$\cdot 770$ augmented images, where the factor '2' accounts for the inclusion of one additional frame from both sides of the sequence's center frame.

In total, 40 models were trained, with the number of training images for each model and augmentation factor shown in Table 3. This systematic approach enables a detailed evaluation of how increasing dataset size impacts model performance and identifies the most effective augmentation strategy for semantic cloud segmentation

Table 3: Number of training images per augmentation factor for each model

Augmentation Factor	**Training Images (per model)**
0	770
1	2310
2	3850
3	5390
4	6930
5	8470
6	10010
7	11550
8	13090
9	14630
10	16170

Training / Validation Split

A common technique in machine learning and deep learning is to split the dataset into a training and a validation part. This is necessary so that after training the models on the split training data, their performance progress can be evaluated on data the model

hasn't seen before. For this training, the dataset is divided with a fixed split into 80% training data and 20% validation samples. If the human annotated dataset isn't augmented with the automatically annotated data, as described in chapter 3.3, this results in 616 training samples and 154 validation samples. When using augmented data for training, it was decided to exclude whole sequences from training if their base centre frame was part of the validation split. This is because the frames within the 21-frame sequences have strong similarities due to the relatively slow dynamics in adjacent images of ASI sequences. This would give the model an inappropriately large amount of prior information that wouldn't match its usual performance on completely unseen scenes. Excluding these sequences ensures a fair comparison between the models.

Hyperparameter for Training
All models are trained under consistent conditions regarding hyperparameters and model architecture. Training is performed for 40 epochs with a batch size of 4 and a learning rate of 1e-4. The model weights are updated using the AdamW optimiser. The training data is used in its original resolution, as its size is already manageable. No further traditional augmentation techniques are applied, since the ASI images are uniform in shape and do not exhibit variations in perspective or colour.
Training durations range from 25 to 175 minutes per model, depending on the dataset size. After completing the 40 epochs, the best-performing checkpoint is selected based on the highest mean IoU achieved on the validation set. This checkpoint represents the fully supervised model for each combination of augmentation factor and VOS model and is subsequently used for evaluation and comparison.

Table 4: Hyperparameters for the fully-supervised model.

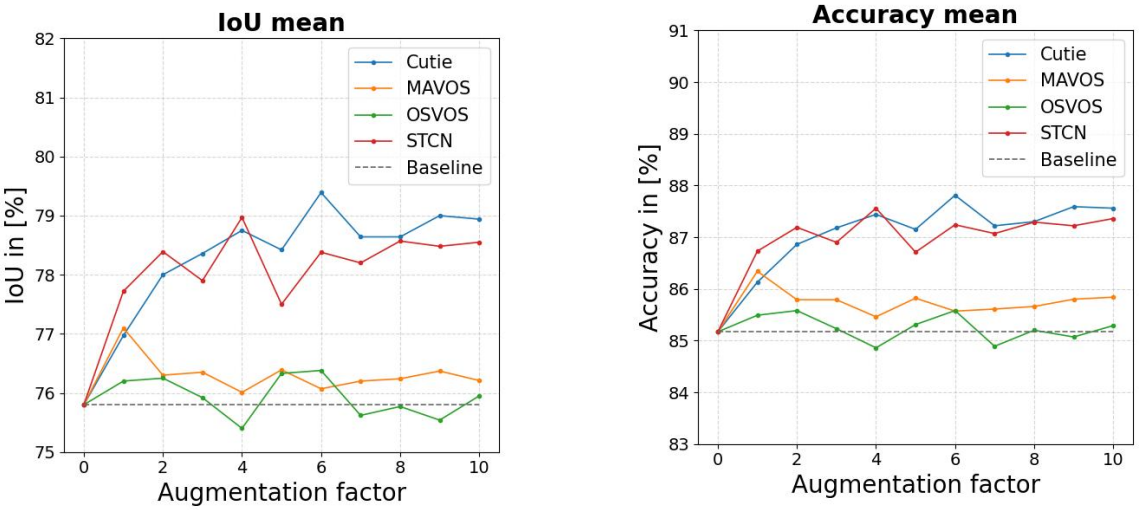
Hyperparameter	**Fully-supervised**
Input size	512x512
Arch., backbone	DeepLabv3+, ResNet50
Initialization	ImageNet
Epochs	40
Batch size	4
Train/Val split	80/20
Optimizer	AdamW
Learning rate	1e-4
Scheduler	OneCycleLR

5.3.1 Quantitative Results

As can be clearly seen the whole procedure of automatically generating new ground truth data proves to be successful. (see figure 27) Almost every combination of an augmentation factor related to the dataset of a specific model leads to an increase in IoU and accuracy compared to the baseline (standard dataset without augmentation). The gain is most pronounced for the Cutie model, which improves IoU by 3.5% points to 79.4% and accuracy by about 2.6% points to 87.8% at a factor of 6, and the STCN model, which performs only slightly less effectively (at augmentation factor 4). Although MAVOS is also able to improve the performance, it falls significantly behind the aforementioned competitors, with the best performance at a factor of 1, increasing the IoU from 75.8% to about 77.1%. As expected, OSVOS is not able to keep up with the modern models, achieving only marginal improvements, and even worse results compared to the baseline at factors 4, 7, 8 and 9 for both IoU and accuracy.
It is possible to draw two conclusions that are applicable to all models. First, increasing the augmentation factor doesn't automatically lead to a higher performance of the resulting model, contrary to intuition. In fact, the largest performance jumps can be observed with augmentation factors 1 and 2, after which only small improvements, if any, can be observed. This is surprising, as images of factors 1 and 2 share many visual features with the existing ground truth image, and thus seem to add very little new information. Though the model can still be improved substantially by these additional ground truth images.
Secondly, the performance gain doesn't follow a monotonically non-decreasing curve, again contrary to what might be expected, but undergoes erratic fluctuations. This is most likely due to the stochastic nature of training deep learning models.

Because cloud segmentation is a multi-class problem, it is useful to look into the metrics of each cloud layer and clear sky separately.
The detection of low-layer clouds currently works best in comparison to mid- or high-layer cloud, with the baseline reaching an IoU value of 65.6% and accuracy of 80.6%. Using data augmentation, IoU values of up to 70.0% and accuracy of 87.8% were achieved with STCN (factor 10) and, surprisingly, OSVOS (factor 10).
Mid-layer cloud detection was until now the biggest weakness of cloud segmentation. Here the baseline only achieved an IoU of 46.0% and accuracy of 59.3%. These values could be increased vastly, specifically by the model STCN increasing the IoU by almost 13% points to 57.8% and the accuracy by 17.2% points to 76.5%.
Similar observations can be made for the high layer class, which is also a difficult class to predict. Here, performance gains for IoU over 9% points to 61.0% were achieved by Cutie (factor 6) and STCN (factor 8), as well as performance gains for accuracy

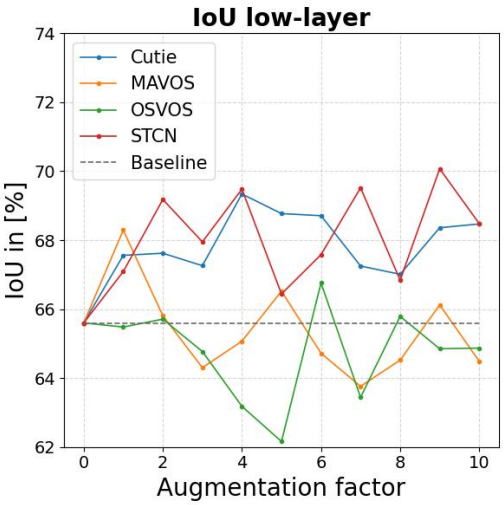
over 11.2% points by OSVOS (factor 10).
Although correctly classifying clouds is certainly the more important task, correctly identifying sky pixels in ASIs is still not negligible. While the baseline already achieves 90.5% IoU and 96.5% accuracy, the best Cutie model increases this to 91.7% IoU and 96.8% accuracy.



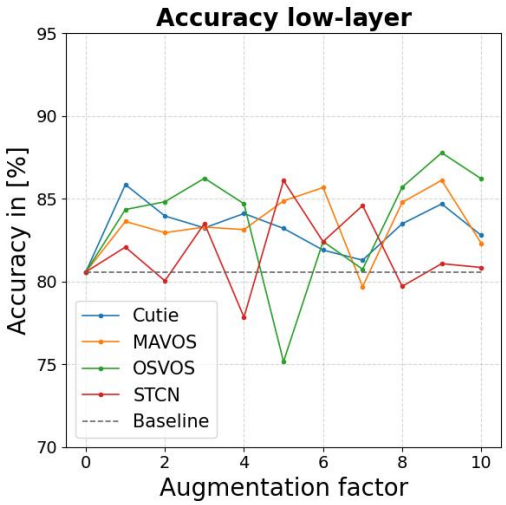
(a) IoU Mean vs. Augmentation Factor

(b) Accuracy Mean vs. Augmentation Factor

Figure 27: Evaluation of IoU and Accuracy Mean with Augmentation Factor.

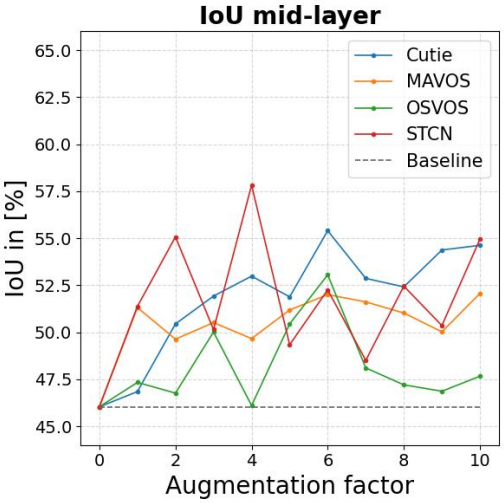


(a) IoU Low-layer vs. Augmentation Factor

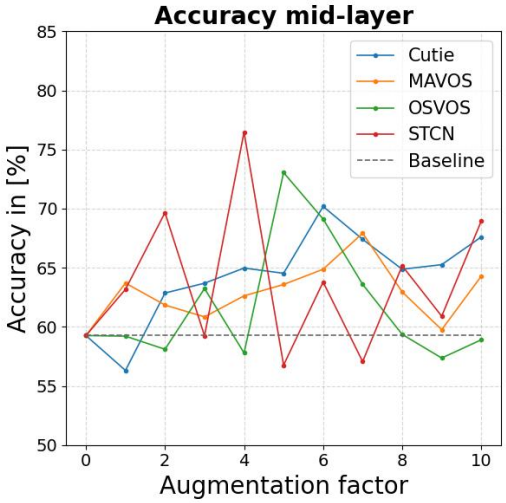


(b) Accuracy Low-layer vs. Augmentation Factor

Figure 28: Evaluation of IoU and Accuracy for the Low-layer with Augmentation Factor.

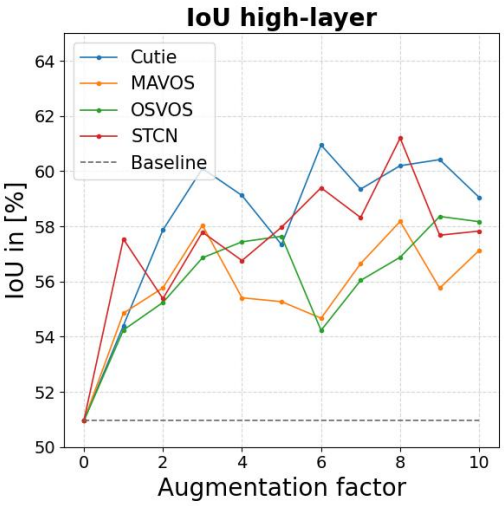


(a) IoU Mid-layer vs. Augmentation Factor

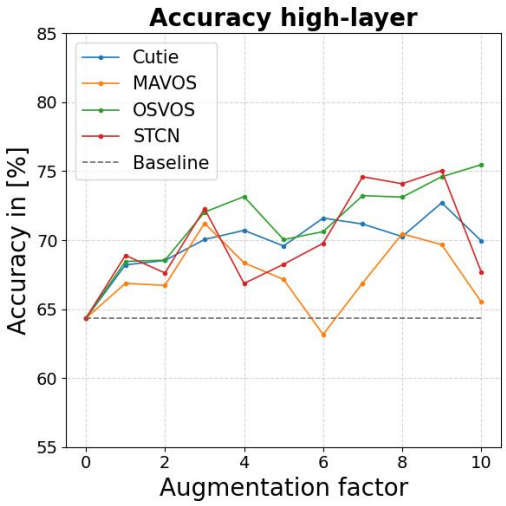


(b) Accuracy Mid-layer vs. Augmentation Factor

Figure 29: Evaluation of IoU and Accuracy for the Mid-layer with Augmentation Factor.

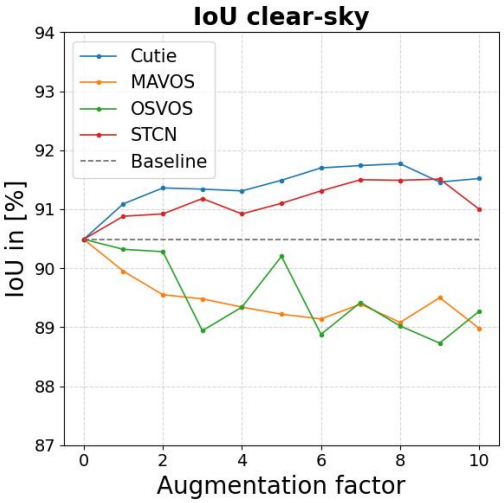


(a) IoU High-layer vs. Augmentation Factor

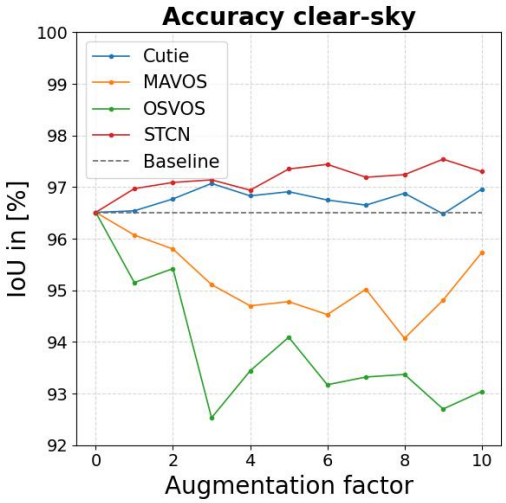


(b) Accuracy High-layer vs. Augmentation Factor

Figure 30: Evaluation of IoU and Accuracy for the High-layer with Augmentation Factor.



(a) IoU Clear Sky vs. Augmentation Factor



(b) Accuracy Clear Sky vs. Augmentation Factor

Figure 31: Evaluation of IoU and Accuracy for Clear Sky with Augmentation Factor.

For further investigations the confusion matrices of each model with its best factor

will be investigated, the best models being determined by the highest value accuracy mean and IoU mean values. These metrics yield the following result:

- Cutie: factor 6
- STCN: factor 4
- MAVOS: factor 1
- OSVOS: factor 6

The confusion matrices for the baseline and for each of the best four models are shown in figure 32. All values are normalised by the respective class, i.e. divided by the sum of each row.
In the confusion matrix of the current cloud segmentation models (baseline) the following major weaknesses can be observed:

- of all actual mid-layer pixels, only 59% were correctly predicted as mid-layer
- of all actual mid-layer pixels, 23% were misclassified as low-layer
- of all actual high-layer pixels, only 64% were correctly predicted as high-layer
- of all actual high-layer pixels, 22% were misclassified as sky

As can be seen in the figures 32, utilizing the newly created dataset considerably helps to address these shortcomings of the baseline model. The Cutie dataset increases the true-positive value of the mid-layer class by 11% points to 70% and of the high-layer class by 8% points to 72%, while also reducing the aforementioned misclassification in the mid- and high-layer classes by 4% points each. Akin is noticeable for the OSVOS dataset, but to a lesser extent, achieving 69% recall for the mid-layer and 71% for the high layer class. The STCN achieves remarkable 76% recall for the mid-layer class but struggles to maintain the true-positive value for the low-layers. Finally MAVOS achieves a high true-positive value of 71% for the high-layer class, but at the same time suffers a loss in recall for the sky- and low-layer classes.

At last, table 5 compares the performance of the binary segmentation - distinguishing only between cloud and sky. While the baseline already achieves remarkable values for both accuracy and IoU, the Cutie dataset can also improve the performance for this setting, increasing the binary accuracy by 0.9%-points to 94.9% and the IoU by almost 1.5%-points to 90.3%. The STCN can also slightly improve its performance for binary segmentation, while OSVOS and MAVOS suffer losses.

Table 5: Binary performance comparison of the different models.

Metric	Baseline	Cutie	STCN	MAVOS	OSVOS
Accuracy binary	0.9399	**0.9486**	0.9426	0.9369	0.9316
IoU binary	0.8881	**0.9028**	0.8926	0.8820	0.8733

Therefore, it appears that the most efficient new ground truth data was generated by the Cutie model with an augmentation factor of 6. In the appendix the entire metrics for all models and every factor are attached.

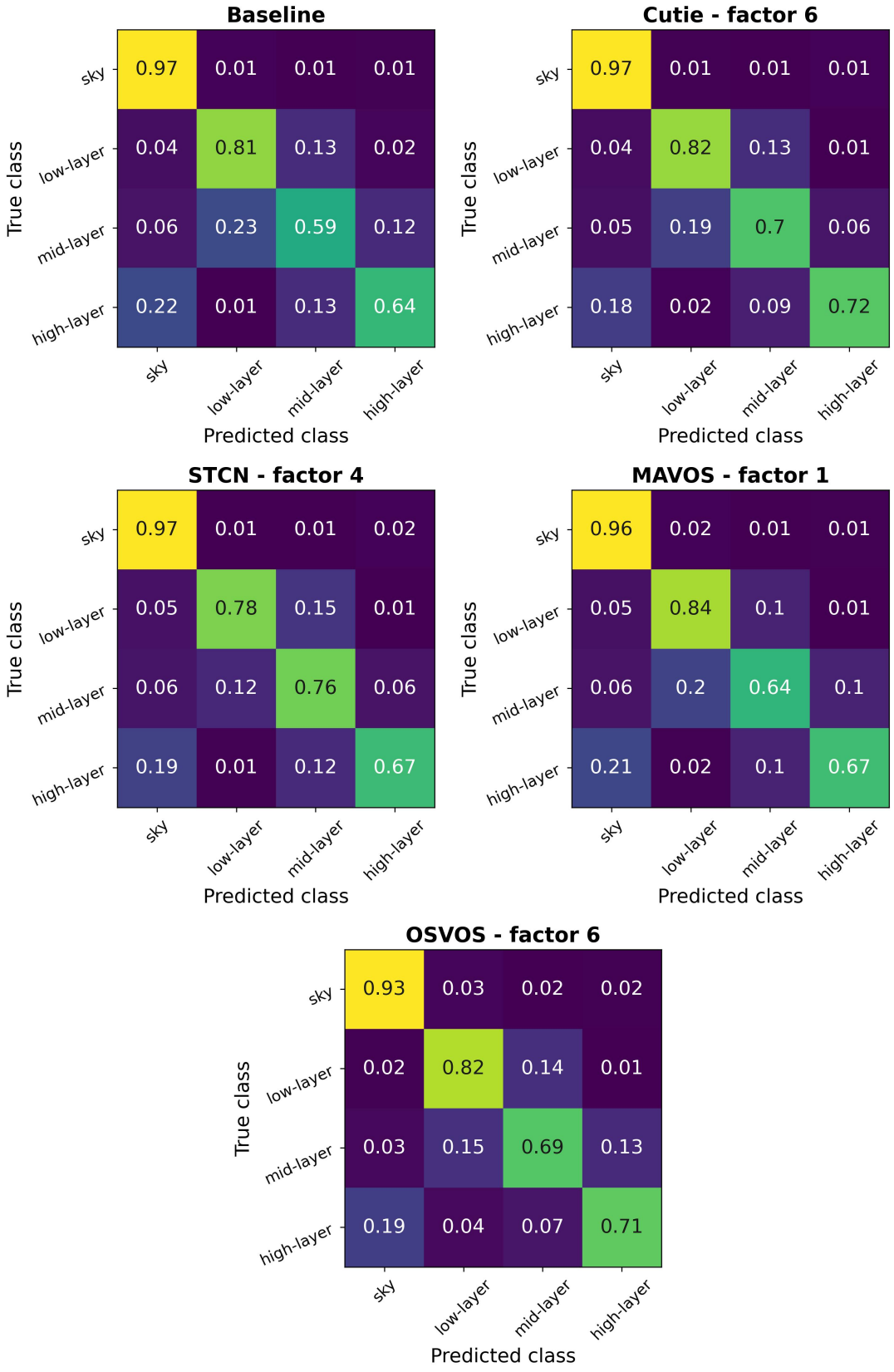


Figure 32: Confusion matrices for the differently trained models

5.3.2 Qualitative Results

With the aid of the confusion matrices in the section above it could be seen that especially the Cutie dataset was able to achieve better distinguishing between the mid- and low-layer clouds. This can also be seen in the first column of figure 33 were the baseline struggled to correctly classify the present mid-layer clouds, it can be clearly seen that through the datasets of Cutie and OSVOS the distinction could be accomplished.

The second column shows a scenario with very fine high layer clouds close to the solar disk, in addition to some scattered low layer clouds. Again, the baseline completely fails to identify this part as clouds and simply classifies it as sky instead. All of the newly generated data sets were able to overcome this problem and recognise the presence of high layer clouds, especially Cutie and OSVOS.

In the third column the baseline models predicts quite noisy results, for a seemingly simple condition with only low-layer clouds, but also predicts large parts of mid-layer and even a few high-layer clouds. While OSVOS even further amplifies the total noise in its prediction, the clear superiority of Cutie is evident. It is able to significantly minimise the noise in the predictions, although there are some high layer classifications around the solar disk.

In the fourth column of figure 33 the advantage of Cutie is further confirmed. Its dataset is the only one that doesn't misidentify low-level clouds in the lower part of the ASI, but correctly identifies only mid- and high-level clouds.

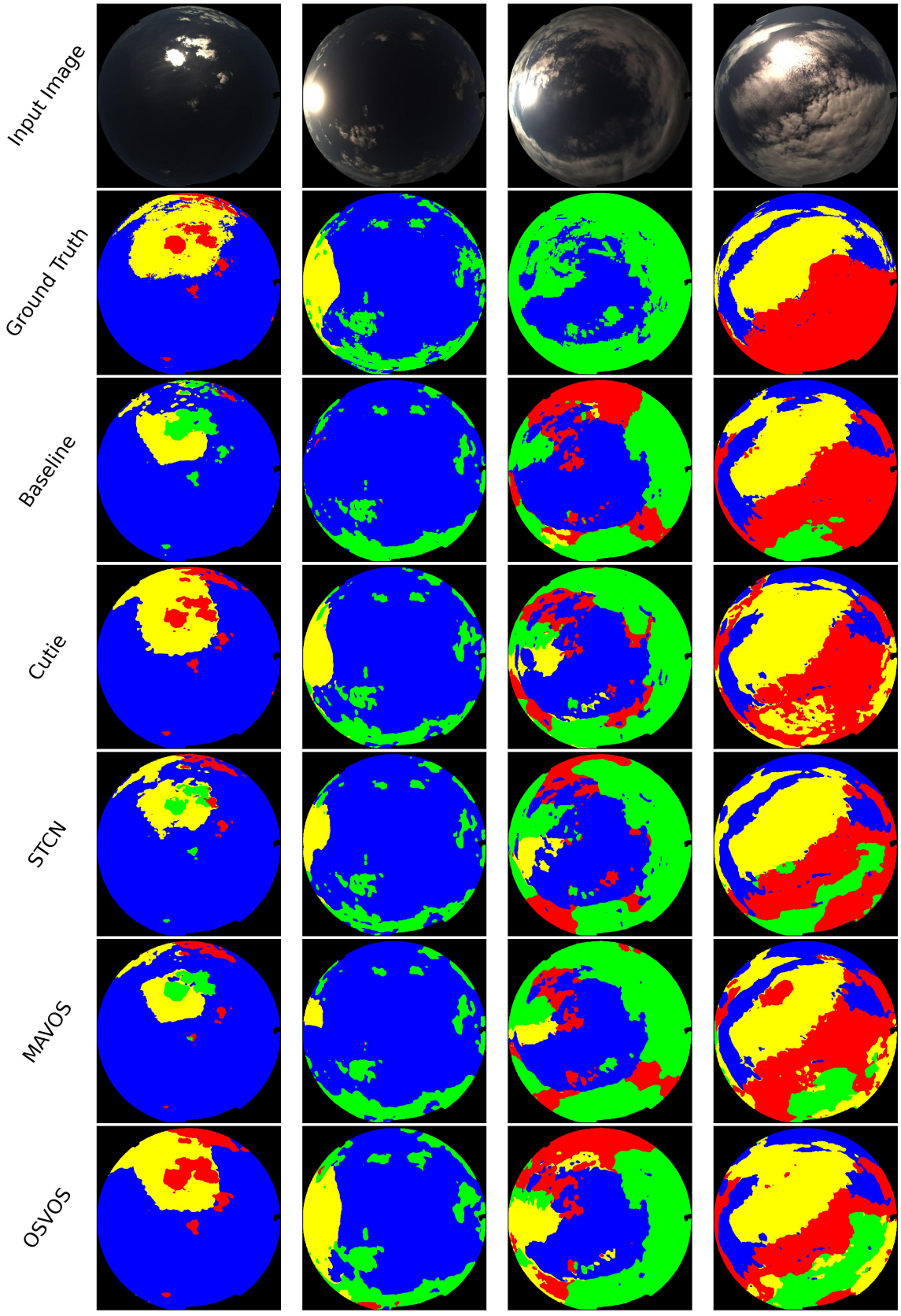


Figure 33: Examples from the models on the validation data trained on the different newly generated datasets (with the best factor each) and the baseline. Color coding: black (background), blue (sky), yellow (high-layer clouds), red (mid-layer clouds), green (low-layer clouds).

5.4 Evaluation of the implemented motion-cued cloud segmentation models

The aim of this evaluation is not only to assess the usefulness of the general idea of incorporating motion cues for semantic cloud segmentation, but also to compare the two different model architectures, namely the stacked and the two-stream approach. Furthermore, the effect of the post-processing on the estimated optic flow maps will be investigated.

Training routine
A total of six models have been trained, allowing a comprehensive conclusion on the efficacy of the techniques applied:

- Stacked model, without post-processing of the optical flow maps
- Stacked model, with post-processing of the optical flow maps
- Two-stream model, without post-processing of the optical flow maps
- Two-stream model, with post-processing of the optical flow maps

Depending on the results of the post-processing chain to the final segmentation result, a final stacked and two-stream model will be trained either including post-processing or not, including the newly created dataset presented in chapter 3.3.

Hyperparameters, Hardware and Metrics
The hyperparameters remain identical to those presented in 5.3, except that the number of epochs is increased to 60 epochs, since training takes considerably longer to converge with increasing input size due to the additional optical flow maps.
Training was performed on two RTX 3090s and took about 3 hours without post-processing, 4 hours with post-processing, and about 22 hours for the final models with an augmentation factor of 6 for the Cutie created dataset.
Since the underlying task of semantic cloud segmentation remains the same, the metrics used are still accuracy, IoU, precision and recall.

Table 6: Performance of motion-cue models trained on the standard dataset
Configurations: Conf 1: Stacked model, no postprocessing
Conf 2: Stacked model, with postprocessing
Conf 3: Two-stream model, no postprocessing
Conf 4: Two-stream model, with postprocessing

	Baseline	Motion-cue models			
	not augmented	**Conf 1**	**Conf 2**	**Conf 3**	**Conf 4**
accuracy	0.8505	0.8484	0.8185	**0.8533**	0.7059
accuracy_binary	0.9400	0.9392	0.9243	**0.9430**	0.7464
accuracy:sky	0.9639	0.9686	0.9406	**0.9753**	0.9577
accuracy:low-layer	**0.8221**	0.7697	0.7759	0.7969	0.6730
accuracy:mid-layer	0.5878	**0.6297**	0.4790	0.6058	0.1775
accuracy:high-layer	0.6165	0.6047	**0.6605**	0.5996	0.0669
iou_mean	0.7558	0.7531	0.7158	**0.7591**	0.5254
iou:sky	0.9050	0.9037	0.8792	**0.9090**	0.6925
iou:low-layer	0.6443	0.6400	0.6105	**0.6639**	0.5300
iou:mid-layer	0.4562	**0.4704**	0.3707	0.4624	0.1697
iou:high-layer	**0.5131**	0.4837	0.4437	0.4818	0.0634
iou_binary_mean	0.8883	0.8861	0.8603	**0.8921**	0.5842

Table 7: Performance of motion-cue models trained on the best augmented dataset ("Cutie - factor 6")
Configurations: Conf 5: Stacked model, no postprocessing
Conf 6: Two-stream model, no postprocessing

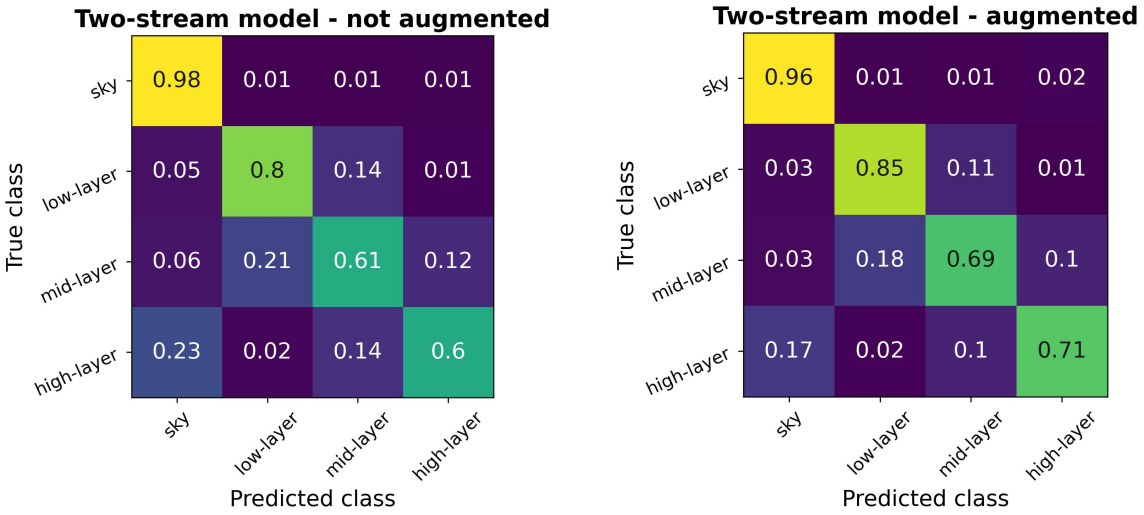
	Baseline	Motion-cue models	
	augmented	**Conf 5**	**Conf 6**
accuracy	**0.8781**	0.8608	0.8775
accuracy_binary	0.9486	0.9459	**0.9490**
accuracy:low-layer	0.8190	0.8315	**0.8520**
accuracy:mid-layer	**0.7018**	0.5889	0.6873
accuracy:high-layer	**0.7160**	0.6787	0.7118
iou_mean	0.7939	0.7700	**0.7942**
iou:low-layer	0.6871	0.6710	**0.7147**
iou:mid-layer	**0.5540**	0.4781	0.5473
iou:high-layer	**0.6095**	0.5299	0.5755
iou_binary_mean	0.9028	0.8981	**0.9036**

Evaluation
Table 6 presents the results for the motion-cue models trained on the standard dataset, which comprises 770 human-annotated masks. The first column shows the baseline, while the subsequent four columns list the metrics for the stacked and two-stream models, each trained both without and with post-processing of the estimated optical flow maps.
A key observation is that the proposed post-processing steps do not appear to be beneficial. In the two-stream approach, they even lead to a significant deterioration—mean IoU decreases by 23% points compared to the baseline and by 4% for the stacked model. The main rationale for introducing post-processing was to better distinguish sky from cloud pixels, yet the IoU:sky metric and overall accuracy also fall behind the baseline when post-processing is applied. Consequently, these post-processed models are excluded from further analysis and the final models are trained without the post-processing steps.
Moreover, the results indicate that the two-stream model generally outperforms the stacked model. Specifically, the two-stream model achieves a mean IoU of 75.9% and an accuracy of 85.3%, whereas the stacked model reaches 75.3% and 84.8%, respectively. Notably, the two-stream model even surpasses the baseline by 0.3% points for both metrics, albeit by a relatively small margin, thereby demonstrating the effectiveness of enriching semantic cloud segmentation with motion cues. By contrast, the stacked model falls just short of the baseline (0.3% points in IoU and 0.2% points in accuracy).

Table 7 then shows the results of the final models, which represent the synthesis of this thesis's achievements. The findings observed for the non-augmented models are reinforced: the two-stream model remains clearly superior to the stacked model, posting an accuracy of 87.8% and an IoU of 79.2%, compared to 86.1% and 77.0% for the stacked model. At the same time, the stacked model no longer keeps pace with the baseline that was trained on augmented data.
When comparing the two-stream model with the baseline of table 7, the results are effectively head-to-head: accuracy and IoU are nearly identical, and any differences alternate slightly in favor of one model or the other. A closer look reveals that the two-stream model detects low-layer clouds more effectively, increasing the respective accuracy and IoU by 3.3% points and 2.7% points, but lags behind the baseline for mid- and high-layer clouds.

Lastly, all three models trained on the augmented dataset outperform every model trained on the standard dataset, confirming once again the effectiveness of the data augmentation strategies introduced in Chapter 3.3.

These observations are supported by the confusion matrices in figure 34, where along the main diagonal the advantage of the model trained on the augmented dataset is evident, as the true positive rates for the different cloud layers are markedly higher, while the misclassifications are attenuated. Compared to the best dataset configuration - Cutie, factor 6 - shown in the confusion matrix in figure 32, the best two-stream model can increase the recall for the low-layer class by 3% points to 85%, while incurring only small losses of 1% points in the mid- and high-layers.



(a) Confusion matrix for the two-stream motion-cue model, trained without augmentation

(b) Confusion matrix for the two-stream motion-cue model, trained with augmentation

Figure 34: Comparison of confusion matrices for the two-stream motion-cue model trained with and without augmentation.

6 Conclusion and outlook

6.1 Conclusion

The primary objective of this thesis was to enhance semantic cloud segmentation models. The key contributions of this work are as follows:

- The existing human-annotated ground truth dataset was extended using semi-supervised video segmentation. Four different models were benchmarked, ultimately expanding the dataset to 16.170 image-mask pairs.

- Extensive experiments were conducted to determine the optimal dataset configuration—evaluating both the choice of the model and the augmentation factor—for training segmentation models.
- A novel semantic cloud segmentation approach was developed that incorporates motion cues extracted from estimated optical flow maps alongside the regarded images.
  - Two architectural approaches were explored: a simple stacked model and a more refined two-stream model.
  - The impact of manually post-processing optical flow maps before feeding them into the models was also investigated.
- A final two-stream model, trained on the newly created Cutie dataset, ensuring a performance on par or surpassing that of the baseline

Both dataset augmentation and the newly proposed motion-cue model demonstrated success in improving cloud segmentation performance.
The data augmentation approach almost consistently yielded superior results across all models and augmentation factors, confirming that semi-supervised video segmentation is a valuable technique. Notably, the Cutie model with an augmentation factor of six produced the best dataset, improving accuracy by 2.6% points and IoU by 3.5% points. Additionally, significant improvements were achieved in distinguishing different cloud layers. In particular, the recall for mid-layer clouds increased by an impressive 11% points, and for high-layer clouds by 8% points, with their respective IoU scores rising by 9.4% points and 10% points.
For many applications, the primary objective is to differentiate between sky and cloud pixels. Even in this binary setting, the IoU improved by 1.5% points, reaching 90.3%.

The approach of combining spatial and motion information for cloud detection has not yet been discussed in the literature, so the results are auspicious for further research in this area. It has been shown that the implemented two-stream approach substantially outperforms the naive stacked model and can already outperform the SOTA model in the metrics of IoU, setting a new SOTA of 79.42% with the best final model, and notably improves the segmentation performance for the low-level clouds by 2.8% points for the IoU to 70.5% and the accuracy by 3.3% points to 85.2%. Furthermore, it was shown that the developed post-processing doesn't aid the segmentation process, contrary to what was expected.
These results have laid the foundation for the integration of the temporal dimension in cloud detection and motivate further research in this area.

6.2 Outlook

Further research on learning better feature representations by integrating motion information into the segmentation process is highly recommended.
Although unsupervised optical flow estimation yields practical results, even better performance can be expected with supervised learning, given the availability of perfect ground-truth data. This would also enable the use of more advanced models, such as those based on transformer architectures or video optical flow models. [Hua+22; Shi+23] Supervised training would require ground-truth flow maps for both training and validation, which could be obtained through synthetic data generation using tools like *Blender* or *kubris* with *Vector Pass*. This approach also presents an opportunity to generate highly specific scenarios that are currently underrepresented in existing datasets, ultimately leading to more balanced data distributions.

Video segmentation is another promising path, as it closely resembles how humans approach the problem by analysing coherent frame sequences to recognize distinctive long-term movement and dynamic patterns. At present, the use of optical flow limits the range of frames considered to two, which doesn't allow long-term developments to be captured. [Zhu+24; Rav+24; Lee+23]

Moreover, the application of interpretable AI techniques (also known as Explainable AI (XAI)) can help identify the reasons for cloud misclassifications, improving both reliability and trust in predictions. By employing gradient-based or perturbation-based methods, a deeper understanding of model behavior and robustness under varying conditions can be achieved. [GTK24]

Finally, unsupervised learning for cloud segmentation—whether in images or videos—remains an area worth revisiting, especially given recent advancements such as Meta's I-JEPA and V-JEPA models. These innovations enhanced the efficiency of utilizing unlabeled data, narrowing the gap between supervised and unsupervised learning. [Bar+24; Ass+23]

Sources and literature

[Alf+24] Andrea Alfarano, Luca Maiano, Lorenzo Papa, and Irene Amerini. "Estimating optical flow: A comprehensive review of the state of the art". In: *Computer Vision and Image Understanding* (2024), p. 104160.

[Alm] Plataforma Solar de Almería. `http://www.psa.es/es/gen/objetivos.php`. Accessed: 2025-04-03.

[Ant+16] Javier Antonanzas et al. "Review of photovoltaic power forecasting". In: *Solar energy* 136 (2016), pp. 78–111.

[Ass+23] Mahmoud Assran et al. "Self-supervised learning from images with a joint-embedding predictive architecture". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 15619–15629.

[Bak+11] Simon Baker et al. "A database and evaluation methodology for optical flow". In: *International journal of computer vision* 92 (2011), pp. 1–31.

[Bar+24] Adrien Bardes et al. "Revisiting feature prediction for learning visual representations from video". In: *arXiv preprint arXiv:2404.08471* (2024).

[Blu+22] Niklas Benedikt Blum et al. "Analyzing Spatial Variations of Cloud Attenuation by a Network of All-Sky Imagers". In: *Remote Sensing* 14.22 (2022). ISSN: 2072-4292. DOI: `10.3390/rs14225685`. URL: `https://www.mdpi.com/2072-4292/14/22/5685`.

[Cae+21] Sergi Caelles et al. "One-Shot Video Object Segmentation". In: (2021).

[Cao+21] Hu Cao et al. *Swin-Unet: Unet-like Pure Transformer for Medical Image Segmentation*. 2021. arXiv: `2105.05537 [eess.IV]`. URL: `https://arxiv.org/abs/2105.05537`.

[Che+16] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. "Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs". In: (2016).

[Che+17] Jingchun Cheng, Yi-Hsuan Tsai, Shengjin Wang, and Ming-Hsuan Yang. "Segflow: Joint learning for video object segmentation and optical flow". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 686–695.

[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.

[Che+24] Ho Kei Cheng, Seoung Wug Oh, Brian Price, Joon-Young Lee, and Alexander Schwing. “Putting the Object Back into Video Object Segmentation”. In: (2024).

[CS22] Ho Kei Cheng and Alexander G. Schwing. “XMem: Long-Term Video Object Segmentation with an Atkinson-Shiffrin Memory Model”. In: (2022).

[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: (2021).

[CVC23] Gabriela Csurka, Riccardo Volpi, and Boris Chidlovskii. “Semantic Image Segmentation: Two Decades of Research”. In: (2023).

[Din+23] Henghui Ding et al. “MOSE: A New Dataset for Video Object Segmentation in Complex Scenes”. In: (2023).

[DLW15] Soumyabrata Dev, Yee Hui Lee, and Stefan Winkler. “Multi-level semantic labeling of sky/cloud images”. In: *2015 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2015, pp. 636–640.

[Dos+15] Alexey Dosovitskiy et al. “Flownet: Learning optical flow with convolutional networks”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2758–2766.

[Dos+16] Alexey Dosovitskiy et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: (2016).

[Eli+22] Elizar Elizar, Mohd Asyraf Zulkifley, Rusdha Muharar, Mohd Hairi Mohd Zaman, and Seri Mastura Mustaza. “A review on multiscale-deep-learning applications”. In: *Sensors* 22.19 (2022), p. 7384.

[Fab+22] Yann Fabel et al. “Applying self-supervised learning for semantic cloud segmentation of all-sky images”. In: *Atmospheric Measurement Techniques* (2022).

[Fab+24] Yann Fabel et al. “Leveraging generative models for enhanced solar irradiance ramp detection”. In: (2024).

[FPZ16] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. “Convolutional two-stream network fusion for video action recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 1933–1941.

[FT19] William Falcon and The PyTorch Lightning team. *PyTorch Lightning*. Version 1.4. Mar. 2019. DOI: 10.5281/zenodo.3828935. URL: https://github.com/Lightning-AI/lightning.

[Gao+22] Mingqi Gao et al. “Deep learning for video object segmentation: a review”. In: (2022).

[Gar88] Louis Garand. “Automated recognition of oceanic cloud patterns. Part I: Methodology and application to cloud climatology”. In: *Journal of Climate* 1.1 (1988), pp. 20–39.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. `http://www.deeplearningbook.org`. MIT Press, 2016.

[Gib50] James J Gibson. *The perception of the visual world.* Houghton Mifflin, 1950.

[GKF23] Dirk Giggenbach, Marcus T Knopp, and Christian Fuchs. “Link budget calculation in optical LEO satellite downlinks with on/off-keying and large signal divergence: A simplified methodology”. In: *International Journal of Satellite Communications and Networking* 41.5 (2023), pp. 460–476.

[Gre+22] Klaus Greff et al. “Kubric: A scalable dataset generator”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 3749–3761.

[GTK24] Rokas Gipiškis, Chun-Wei Tsai, and Olga Kurasova. “Explainable AI (XAI) in image segmentation in medicine, industry, and beyond: A survey”. In: *ICT Express* (2024).

[Has+20] Marcel Hasenbalg, Pascal Kuhn, Stefan Wilbert, Bijan Nouri, and A. Kazantzidis. “Benchmarking of six cloud segmentation algorithms for ground-based all-sky imagers”. In: *Solar Energy* (2020).

[He+15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep Residual Learning for Image Recognition”. In: (2015).

[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.

[HMS10] Anna Heinle, Andreas Macke, and Anand Srivastav. “Automatic cloud classification of whole sky images”. In: *Atmospheric Measurement Techniques* 3.3 (2010), pp. 557–567.

[Hon+22] Lingyi Hong et al. “LVOS: A Benchmark for Long-term Video Object Segmentation”. In: (2022).

[HRW01] Carole J Hahn, William B Rossow, and Stephen G Warren. “ISCCP cloud properties associated with standard cloud types identified in individual surface observations”. In: *Journal of Climate* 14.1 (2001), pp. 11–28.

[HS81] Berthold KP Horn and Brian G Schunck. “Determining optical flow”. In: *Artificial intelligence* 17.1-3 (1981), pp. 185–203.

[Hua+22] Zhaoyang Huang et al. “Flowformer: A transformer architecture for optical flow”. In: *European conference on computer vision*. Springer. 2022, pp. 668–685.

[Ilg+17] Eddy Ilg et al. “Flownet 2.0: Evolution of optical flow estimation with deep networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2462–2470.

[Jad20] Shruti Jadon. “A survey of loss functions for semantic segmentation”. In: (2020).

[Jon+20] Rico Jonschkowski et al. “What Matters in Unsupervised Optical Flow”. In: *arXiv preprint arXiv:2006.04902* (2020).

[JRC15] Vijai T Jayadevan, Jeffrey J Rodriguez, and Alexander D Cronin. “A new contrast-enhancing feature for cloud detection in ground-based sky images”. In: *Journal of Atmospheric and Oceanic Technology* 32.2 (2015), pp. 209–219.

[Kaz+12] Andreas Kazantzidis, Panagiotis Tzoumanikas, Alkiviadia F Bais, Spiros Fotopoulos, and George Economou. “Cloud detection and classification with the use of whole-sky ground-based images”. In: *Atmospheric Research* 113 (2012), pp. 80–88.

[Kho+16] Anna Khoreva, Federico Perazzi, Rodrigo Benenson, Bernt Schiele, and Alexander Sorkine-Hornung. *Learning Video Object Segmentation from Static Images*. 2016. arXiv: 1612.02646 [cs.CV]. URL: https://arxiv.org/abs/1612.02646.

[Kho+19] Anna Khoreva, Rodrigo Benenson, Eddy Ilg, Thomas Brox, and Bernt Schiele. *Lucid Data Dreaming for Video Object Segmentation*. 2019. arXiv: 1703.09554 [cs.CV]. URL: https://arxiv.org/abs/1703.09554.

[Kim+20] Dahun Kim, Sangyhun Woo, Joon-Young Lee, and In So Kweon. “Video Panoptic Segmentation”. In: (2020).

[Kir+23] Alexander Kirillov et al. *Segment Anything*. 2023. arXiv: 2304.02643 [cs.CV]. URL: https://arxiv.org/abs/2304.02643.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: (2012).

[Lee+23] Minhyeok Lee, Suhwan Cho, Seunghoon Lee, Chaewon Park, and Sangyoun Lee. "Unsupervised video object segmentation via prototype memory network". In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2023, pp. 5924–5934.

[LH17] Ilya Loshchilov and Frank Hutter. "Decoupled Weight Decay Regularization". In: (2017).

[LK81] Bruce D Lucas and Takeo Kanade. "An iterative image registration technique with an application to stereo vision". In: *IJCAI'81: 7th international joint conference on Artificial intelligence*. Vol. 2. 1981, pp. 674–679.

[LLY11] Qingyong Li, Weitao Lu, and Jun Yang. "A hybrid thresholding algorithm for cloud detection on ground-based color images". In: *Journal of atmospheric and oceanic technology* 28.10 (2011), pp. 1286–1296.

[Lon+06] Charles N Long, Jeff M Sabburg, Josep Calbó, and David Pagès. "Retrieving cloud characteristics from ground-based daytime color all-sky images". In: *Journal of Atmospheric and Oceanic Technology* 23.5 (2006), pp. 633–652.

[LSD15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully Convolutional Networks for Semantic Segmentation". In: (2015).

[Mag+25] David Mageira et al. "Advancing Semantic Cloud Segmentation in All-Sky Images: A Semi-Supervised Learning Approach with Ceilometer-Driven Weak Labels". In: *Solar Energy* (2025).

[May+18] Nikolaus Mayer et al. "What makes good synthetic training data for learning disparity and optical flow estimation?" In: *International Journal of Computer Vision* 126 (2018), pp. 942–960.

[MBS02] Bertalmio Marcelo, Andrea L. Bertozzi, and Guillermo Sapiro. "Navier-Stokes, Fluid Dynamics, and Image and Video Inpainting". In: (2002).

[MHR18] Simon Meister, Junhwa Hur, and Stefan Roth. "Unflow: Unsupervised learning of optical flow with a bidirectional census loss". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 32. 1. 2018.

[NHH15] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. *Learning Deconvolution Network for Semantic Segmentation*. 2015. arXiv: 1505.04366 [cs.CV]. URL: https://arxiv.org/abs/1505.04366.

[Nic+22] Nicki Skafte Detlefsen et al. *TorchMetrics - Measuring Reproducibility in PyTorch*. Feb. 2022. DOI: 10.21105/joss.04101. URL: https://github.com/Lightning-AI/torchmetrics.

[Nou+19] Bijan Nouri et al. “Determination of cloud transmittance for all sky imager based solar nowcasting”. In: *Solar Energy* 181 (2019), pp. 251–263.

[Nou+23] Bijan Nouri et al. “Probabilistic solar nowcasting based on all-sky imagers”. In: *Solar Energy* 253 (2023), pp. 285–307. ISSN: 0038-092X. DOI: https://doi.org/10.1016/j.solener.2023.01.060. URL: https://www.sciencedirect.com/science/article/pii/S0038092X23000683.

[Oh+19] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. *Video Object Segmentation using Space-Time Memory Networks*. 2019. arXiv: 1904.00607 [cs.CV]. URL: https://arxiv.org/abs/1904.00607.

[Org17] W. M. Organization. *International Cloud Atlas*. World Meteorological Organiza- tion, 2017.

[Pas+19] Adam Paszke et al. “Pytorch: An imperative style, high-performance deep learning library”. In: *Advances in neural information processing systems* 32 (2019).

[Pen+15] Zhenzhou Peng et al. “3D cloud detection and tracking system for solar forecast using multiple sky imagers”. In: *Solar Energy* 118 (2015), pp. 496–519.

[Per+16] F. Perazzi et al. “A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation”. In: (2016).

[Pon+17] Jordi Pont-Tuset et al. “The 2017 DAVIS Challenge on Video Object Segmentation”. In: (2017).

[Rav+24] Nikhila Ravi et al. *SAM 2: Segment Anything in Images and Videos*. 2024. arXiv: 2408.00714 [cs.CV]. URL: https://arxiv.org/abs/2408.00714.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV]. URL: https://arxiv.org/abs/1505.04597.

[Roh25] Chris Rohlfs. “Generalization in neural networks: A broad survey”. In: *Neurocomputing* 611 (2025), p. 128701.

[Sha+24] Abdelrahman Shaker et al. “Efficient Video Object Segmentation via Modulated Cross-Attention Memory”. In: (2024).

[Shi+17] Jae Shin Yoon et al. “Pixel-level matching for video object segmentation using convolutional neural networks”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2167–2176.

[Shi+23] Xiaoyu Shi et al. “Videoflow: Exploiting temporal cues for multi-frame optical flow estimation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 12469–12480.

[SK19] Connor Shorten and Taghi M. Khoshgoftaar. “A survey on Image Data Augmentation for Deep Learning”. In: (2019).

[Sol+09] Albert Solé-Benet, Yolanda Cantón, Roberto Lázaro, and Juan Puigdefábregas. “Meteorización y erosión en el subdesierto de Tabernas”. In: (2009).

[SRB10] Deqing Sun, Stefan Roth, and Michael J Black. “Secrets of optical flow estimation and their principles”. In: *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE. 2010, pp. 2432–2439.

[STB15] K. Shields, J. Tovar-Pescador, and F. J. Batlles. “Performance optimization of concentrating solar power plants through use of direct normal irradiance forecasting”. In: *Solar energy* 75 (2015), pp. 518–524.

[Ste+21] Andreas Steiner et al. “How to train your vit? data, augmentation, and regularization in vision transformers”. In: *arXiv preprint arXiv:2106.10270* (2021).

[Sto+21] Austin Stone, Daniel Maurer, Alper Ayvaci, Anelia Angelova, and Rico Jonschkowski. “SMURF: Self-Teaching Multi-Frame Unsupervised RAFT with Full-Image Warping”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021.

[SZ14] Karen Simonyan and Andrew Zisserman. “Two-stream convolutional networks for action recognition in videos”. In: *Advances in neural information processing systems* 27 (2014).

[Tar+14] Alireza Taravat, Fabio Del Frate, Cristina Cornaro, and Stefania Vergari. “Neural networks and support vector machine algorithms for automatic cloud classification of whole-sky ground-based images”. In: *IEEE Geoscience and remote sensing letters* 12.3 (2014), pp. 666–670.

[TAS17] Pavel Tokmakov, Karteek Alahari, and Cordelia Schmid. *Learning Video Object Segmentation with Visual Memory*. 2017.

[TD20] Zachary Teed and Jia Deng. “Raft: Recurrent all-pairs field transforms for optical flow”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer. 2020, pp. 402–419.

[Tel03] Alexandru Telea. “An Image Inpainting Technique Based on the Fast Marching Method”. In: (2003).

[Voi+19] Paul Voigtlaender 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.

[Wei+13] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. “DeepFlow: Large displacement optical flow with deep matching”. In: *Proceedings of the IEEE international conference on computer vision*. 2013, pp. 1385–1392.

[Wen+20] Haoran Wen et al. “Deep learning based multistep solar forecasting for PV ramp-rate control using sky images”. In: *IEEE Transactions on Industrial Informatics* 17.2 (2020), pp. 1397–1406.

[WRZ00] Junhong Wang, William B Rossow, and Yuanchong Zhang. “Cloud vertical structure and its variations from a 20-yr global rawinsonde dataset”. In: *Journal of climate* 13.17 (2000), pp. 3041–3056.

[Xie+20] Wanyi Xie et al. “SegCloud: A novel cloud image segmentation model using a deep convolutional neural network for ground-based all-sky-view camera observation”. In: *Atmospheric Measurement Techniques* 13.4 (2020), pp. 1953–1961.

[Xu+18] Ning Xu et al. “YouTube-VOS: A Large-Scale Video Object Segmentation Benchmark”. In: (2018).

[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.

[Zha+22] Bowen Zhang et al. *SegViT: Semantic Segmentation with Plain Vision Transformers*. 2022. arXiv: 2210.05844 [cs.CV]. URL: https://arxiv.org/abs/2210.05844.

[Zho+22] Tianfei Zhou, Fatih Porikli, David J. Crandall, Luc Van Gool, and Wenguan Wang. “A Survey on Deep Learning Technique for Video Segmentation”. In: (2022).

[Zhu+24] Yunzhi Zhuge, Hongyu Gu, Lu Zhang, Jinqing Qi, and Huchuan Lu. “Learning Motion and Temporal Cues for Unsupervised Video Object Segmentation”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2024).

List of Figures

List of Tables

7 Appendix

Table 8: Evaluation results of the baseline and the Cutie models

	baseline	Cutie									
		1	2	3	4	5	6	7	8	9	10
accuracy	0.8517	0.8613	0.8686	0.8718	0.8744	0.8715	0.8781	0.8722	0.873	0.8759	0.8756
accuracy:sky	0.9651	0.9654	0.9677	0.9707	0.9683	0.9691	0.9675	0.9665	0.9688	0.9648	0.9696
accuracy:low-layer	0.8058	0.8586	0.8396	0.8324	0.8411	0.8321	0.819	0.8129	0.8349	0.8469	0.8279
accuracy:mid-layer	0.5926	0.5631	0.6286	0.6369	0.6497	0.6453	0.7018	0.674	0.6487	0.6526	0.676
accuracy:high-layer	0.6434	0.6821	0.6853	0.7005	0.7071	0.6958	0.716	0.7117	0.7026	0.7271	0.6995
precision	0.8481	0.8587	0.8659	0.8693	0.8717	0.8684	0.8766	0.8705	0.8711	0.8745	0.873
precision:sky	0.9356	0.9417	0.9423	0.9393	0.9412	0.9424	0.9461	0.9475	0.9457	0.9462	0.9423
precision:low-layer	0.7792	0.7601	0.7765	0.778	0.798	0.7985	0.8102	0.7956	0.7724	0.78	0.7983
precision:mid-layer	0.6735	0.7358	0.7184	0.7376	0.7418	0.7257	0.7245	0.7102	0.7319	0.7651	0.7399
precision:high-layer	0.71	0.7287	0.7881	0.8089	0.7828	0.7652	0.8037	0.7814	0.8078	0.7813	0.7912
recall	0.8517	0.8613	0.8686	0.8718	0.8744	0.8715	0.8781	0.8722	0.873	0.8759	0.8756
recall:sky	0.9651	0.9654	0.9677	0.9707	0.9683	0.9691	0.9675	0.9665	0.9688	0.9648	0.9696
recall:low-layer	0.8058	0.8586	0.8396	0.8324	0.8411	0.8321	0.819	0.8129	0.8349	0.8469	0.8279
recall:mid-layer	0.5926	0.5631	0.6286	0.6369	0.6497	0.6453	0.7018	0.674	0.6487	0.6526	0.676
recall:high-layer	0.6434	0.6821	0.6853	0.7005	0.7071	0.6958	0.716	0.7117	0.7026	0.7271	0.6995
iou_mean	0.758	0.7697	0.78	0.7836	0.7875	0.7842	0.7939	0.7864	0.7864	0.79	0.7894
iou:sky	0.9049	0.9109	0.9136	0.9134	0.9131	0.9149	0.917	0.9174	0.9177	0.9146	0.9152
iou:low-layer	0.656	0.6756	0.6762	0.6726	0.6934	0.6877	0.6871	0.6725	0.6701	0.6836	0.6847
iou:mid-layer	0.4603	0.4684	0.5044	0.5192	0.5298	0.5188	0.554	0.5286	0.5241	0.5437	0.5462
iou:high-layer	0.5095	0.544	0.5787	0.601	0.5912	0.5734	0.6095	0.5935	0.602	0.6042	0.5905
accuracy_binary	0.9399	0.9444	0.9463	0.9458	0.9458	0.9471	0.9486	0.9489	0.949	0.9472	0.9473
iou_binary_mean	0.8881	0.8956	0.8986	0.8979	0.8979	0.9001	0.9028	0.9034	0.9035	0.9004	0.9005
iou-binary:sky	0.9049	0.9109	0.9136	0.9134	0.9131	0.9149	0.917	0.9174	0.9177	0.9146	0.9152
iou-binary:cloud	0.865	0.8745	0.878	0.8765	0.877	0.8796	0.8833	0.8842	0.884	0.8808	0.8801

Table 9: Evaluation results of the baseline and the STCN models

	baseline	STCN									
		1	2	3	4	5	6	7	8	9	10
accuracy	0.8517	0.8673	0.8719	0.869	0.8756	0.8671	0.8724	0.8707	0.8729	0.8722	0.8736
accuracy:sky	0.9651	0.9697	0.9709	0.9714	0.9694	0.9735	0.9744	0.9719	0.9724	0.9754	0.973
accuracy:low-layer	0.8058	0.8208	0.8004	0.8351	0.7784	0.861	0.8241	0.8459	0.7971	0.8108	0.8085
accuracy:mid-layer	0.5926	0.6317	0.6966	0.5924	0.7647	0.5676	0.6379	0.5706	0.6516	0.609	0.6894
accuracy:high-layer	0.6434	0.689	0.6762	0.7228	0.6686	0.6825	0.6977	0.746	0.7409	0.7505	0.677
precision	0.8481	0.8644	0.8692	0.8659	0.8763	0.8659	0.8689	0.8677	0.8697	0.8691	0.8708
precision:sky	0.9356	0.9353	0.9347	0.9369	0.9361	0.9342	0.9356	0.9398	0.9393	0.9367	0.9336
precision:low-layer	0.7792	0.786	0.836	0.7848	0.8662	0.7442	0.7897	0.796	0.8055	0.8376	0.8174
precision:mid-layer	0.6735	0.7333	0.7246	0.7653	0.7034	0.7903	0.7429	0.7636	0.7289	0.7439	0.7303
precision:high-layer	0.71	0.7772	0.7536	0.7425	0.7897	0.7937	0.7999	0.7277	0.7786	0.7136	0.7987
recall	0.8517	0.8673	0.8719	0.869	0.8756	0.8671	0.8724	0.8707	0.8729	0.8722	0.8736
recall:sky	0.9651	0.9697	0.9709	0.9714	0.9694	0.9735	0.9744	0.9719	0.9724	0.9754	0.973
recall:low-layer	0.8058	0.8208	0.8004	0.8351	0.7784	0.861	0.8241	0.8459	0.7971	0.8108	0.8085
recall:mid-layer	0.5926	0.6317	0.6966	0.5924	0.7647	0.5676	0.6379	0.5706	0.6516	0.609	0.6894
recall:high-layer	0.6434	0.689	0.6762	0.7228	0.6686	0.6825	0.6977	0.746	0.7409	0.7505	0.677
iou_mean	0.758	0.7772	0.7839	0.779	0.7897	0.775	0.7838	0.782	0.7857	0.7848	0.7855
iou:sky	0.9049	0.9088	0.9092	0.9118	0.9092	0.911	0.9131	0.915	0.9149	0.9151	0.91
iou:low-layer	0.656	0.6709	0.6918	0.6795	0.6948	0.6644	0.6758	0.6952	0.6684	0.7007	0.6848
iou:mid-layer	0.4603	0.5137	0.5507	0.5014	0.5782	0.4933	0.5225	0.485	0.5245	0.5035	0.5495
iou:high-layer	0.5095	0.5754	0.5538	0.5779	0.5676	0.5797	0.594	0.5832	0.612	0.5768	0.5783
accuracy_binary	0.9399	0.9422	0.9428	0.9446	0.9426	0.9439	0.9453	0.9468	0.9467	0.9466	0.9432
iou_binary_mean	0.8881	0.8921	0.8926	0.8957	0.8926	0.8944	0.8969	0.8996	0.8995	0.8992	0.8932
iou-binary:sky	0.9049	0.9088	0.9092	0.9118	0.9092	0.911	0.9131	0.915	0.9149	0.9151	0.91
iou-binary:cloud	0.865	0.869	0.8696	0.8735	0.8697	0.8716	0.8746	0.8784	0.8781	0.8773	0.8701

Table 10: Evaluation results of the baseline and the MAVOS models

	baseline	MAVOS									
		1	2	3	4	5	6	7	8	9	10
accuracy	0.8517	0.8634	0.8579	0.8579	0.8546	0.8582	0.8557	0.8561	0.8566	0.858	0.8584
accuracy:sky	0.9651	0.9607	0.958	0.9511	0.947	0.9478	0.9453	0.9502	0.9407	0.9481	0.9573
accuracy:low-layer	0.8058	0.8363	0.8295	0.8329	0.8314	0.8487	0.8568	0.7969	0.8478	0.8612	0.8231
accuracy:mid-layer	0.5926	0.637	0.6184	0.6084	0.6262	0.6358	0.6487	0.6792	0.6295	0.5976	0.6428
accuracy:high-layer	0.6434	0.6687	0.6672	0.7122	0.6836	0.6715	0.6316	0.6689	0.7046	0.6967	0.6552
precision	0.8481	0.8603	0.8552	0.8575	0.8538	0.8572	0.8578	0.8556	0.858	0.8582	0.8568
precision:sky	0.9356	0.9339	0.9321	0.9379	0.9405	0.9382	0.9399	0.9379	0.9438	0.9411	0.9265
precision:low-layer	0.7792	0.7884	0.7611	0.7382	0.7497	0.7547	0.7258	0.7613	0.7298	0.7401	0.7487
precision:mid-layer	0.6735	0.7251	0.7153	0.7484	0.7057	0.7239	0.7239	0.6824	0.7292	0.7542	0.7325
precision:high-layer	0.71	0.7531	0.7725	0.7582	0.7452	0.7575	0.8026	0.7873	0.7695	0.7363	0.817
recall	0.8517	0.8634	0.8579	0.8579	0.8546	0.8582	0.8557	0.8561	0.8566	0.858	0.8584
recall:sky	0.9651	0.9607	0.958	0.9511	0.947	0.9478	0.9453	0.9502	0.9407	0.9481	0.9573
recall:low-layer	0.8058	0.8363	0.8295	0.8329	0.8314	0.8487	0.8568	0.7969	0.8478	0.8612	0.8231
recall:mid-layer	0.5926	0.637	0.6184	0.6084	0.6262	0.6358	0.6487	0.6792	0.6295	0.5976	0.6428
recall:high-layer	0.6434	0.6687	0.6672	0.7122	0.6836	0.6715	0.6316	0.6689	0.7046	0.6967	0.6552
iou_mean	0.758	0.771	0.763	0.7635	0.7601	0.7639	0.7607	0.762	0.7624	0.7637	0.7621
iou:sky	0.9049	0.8995	0.8955	0.8948	0.8934	0.8922	0.8914	0.8939	0.8908	0.895	0.8898
iou:low-layer	0.656	0.6829	0.6581	0.643	0.6507	0.6652	0.6472	0.6376	0.6452	0.6612	0.6449
iou:mid-layer	0.4603	0.5131	0.4963	0.5051	0.4966	0.5118	0.52	0.5161	0.5102	0.5002	0.5206
iou:high-layer	0.5095	0.5485	0.5577	0.5804	0.5541	0.5527	0.5467	0.5665	0.5818	0.5576	0.5713
accuracy_binary	0.9399	0.9369	0.9344	0.9342	0.9338	0.933	0.9322	0.934	0.9324	0.9348	0.9304
iou_binary_mean	0.8881	0.882	0.8775	0.8776	0.8766	0.875	0.8743	0.8768	0.8743	0.8784	0.8705
iou-binary:sky	0.9049	0.8995	0.8955	0.8948	0.8934	0.8922	0.8914	0.8939	0.8908	0.895	0.8898
iou-binary:cloud	0.865	0.8579	0.8526	0.8539	0.8535	0.8514	0.8508	0.8532	0.8516	0.8555	0.8439

Table 11: Evaluation results of the baseline and the OSVOS models

	baseline	OSVOS									
		1	2	3	4	5	6	7	8	9	10
accuracy	0.8517	0.8549	0.8558	0.8523	0.8486	0.8531	0.8558	0.8489	0.852	0.8507	0.8529
accuracy:sky	0.9651	0.9515	0.9542	0.9253	0.9344	0.9409	0.9317	0.9332	0.9337	0.927	0.9304
accuracy:low-layer	0.8058	0.8435	0.8482	0.8624	0.847	0.7516	0.8242	0.8074	0.857	0.8778	0.8621
accuracy:mid-layer	0.5926	0.5921	0.581	0.6324	0.5779	0.7305	0.6912	0.6359	0.5936	0.5736	0.589
accuracy:high-layer	0.6434	0.6846	0.6855	0.7203	0.7316	0.7004	0.7062	0.7323	0.7312	0.746	0.7547
precision	0.8481	0.8539	0.8541	0.8575	0.8514	0.859	0.8587	0.852	0.8537	0.8552	0.8568
precision:sky	0.9356	0.9467	0.9436	0.9582	0.9531	0.9561	0.9508	0.9554	0.9503	0.9539	0.9566
precision:low-layer	0.7792	0.7454	0.7447	0.7224	0.7131	0.7823	0.7784	0.7475	0.739	0.7128	0.7238
precision:mid-layer	0.6735	0.7023	0.7054	0.7051	0.695	0.6197	0.6952	0.6639	0.6973	0.7192	0.7141
precision:high-layer	0.71	0.7231	0.7399	0.7298	0.7277	0.7651	0.7003	0.7047	0.7192	0.7283	0.7173
recall	0.8517	0.8549	0.8558	0.8523	0.8486	0.8531	0.8558	0.8489	0.852	0.8507	0.8529
recall:sky	0.9651	0.9515	0.9542	0.9253	0.9344	0.9409	0.9317	0.9332	0.9337	0.927	0.9304
recall:low-layer	0.8058	0.8435	0.8482	0.8624	0.847	0.7516	0.8242	0.8074	0.857	0.8778	0.8621
recall:mid-layer	0.5926	0.5921	0.581	0.6324	0.5779	0.7305	0.6912	0.6359	0.5936	0.5736	0.589
recall:high-layer	0.6434	0.6846	0.6855	0.7203	0.7316	0.7004	0.7062	0.7323	0.7312	0.746	0.7547
iou_mean	0.758	0.762	0.7625	0.7592	0.754	0.7633	0.7638	0.7562	0.7577	0.7554	0.7595
iou:sky	0.9049	0.9028	0.8894	0.8934	0.902	0.8888	0.8942	0.8902	0.8873	0.8927	0.8927
iou:low-layer	0.656	0.6548	0.6571	0.6477	0.6318	0.6216	0.6676	0.6344	0.6579	0.6485	0.6487
iou:mid-layer	0.4603	0.4733	0.4676	0.5001	0.461	0.5044	0.5305	0.481	0.472	0.4686	0.4766
iou:high-layer	0.5095	0.5424	0.5524	0.5686	0.5744	0.5764	0.5423	0.5604	0.5688	0.5836	0.5817
accuracy_binary	0.9399	0.9402	0.9397	0.9325	0.9348	0.9401	0.9316	0.9353	0.9328	0.9312	0.9348
iou_binary_mean	0.8881	0.8879	0.887	0.8748	0.8784	0.8879	0.8733	0.8795	0.8748	0.8722	0.8783
iou-binary:sky	0.9049	0.9032	0.9028	0.8894	0.8934	0.902	0.8888	0.8942	0.8902	0.8873	0.8927
iou-binary:cloud	0.865	0.8669	0.8653	0.8547	0.8578	0.8684	0.8518	0.8593	0.8535	0.8514	0.8584