TUM Data Innovation Lab

Munich Data Science Institute (MDSI)

Technical University of Munich

&

ZEISS Group, Chair of Computer Aided Medical Procedures

Final report of project:

Continual Learning for Domain Incremental Semantic Segmentation

Authors	Kristiyan Sakalyan, Gözde Ünver, Anar Abbasli
Mentor(s)	Dr. Ghazal Ghazaei, Alim Bukharaev
TUM Mentor	Dr. Ghazal Ghazaei
Project lead	Dr. Ricardo Acevedo Cabra (MDSI)
Supervisor	Prof. Dr. Massimo Fornasier (MDSI)

Jul 2024

Acknowledgements

We would like to express our gratitude to our supervisors Dr. Ghazal Ghazaei and Alim Bukharaev for their guidance and valuable insights on catastrophic forgetting throughout our study. Our sincere thanks also go to the ZEISS Group for financing this study and providing the necessary infrastructure for our experiments. Additionally, we extend our appreciation to Prof. Dr. Massimo Fornasier, Dr. Ricardo Acevedo Cabra, and the entire team at the TUM Data Innovation Lab for facilitating this collaborative research.

Abstract

In this study, we address the challenge of catastrophic forgetting in domain-incremental semantic segmentation, particularly within the context of medical image analysis. Our focus is on developing methods that enable continual learning, allowing models to adapt to new data without losing previously acquired knowledge. We evaluate and compare various approaches, including regularization-based methods, replay strategies, knowledge distillation techniques, and propose a method based on contrastive learning.
We conduct extensive experiments on two cataract surgery datasets, CaDIS and Cataract-1K, simulating domain shifts typically encountered in medical imaging scenarios. Our findings highlight that contrastive learning, especially when combined with effective sample replay strategies and augmentations like color jittering, significantly mitigates catastrophic forgetting and enhances domain generalization. However, we also observe a trade-off between reducing forgetting and maintaining high performance on new tasks, underscoring the need for balanced strategies.
Our contributions include a comprehensive evaluation of existing and novel methods for continual learning in semantic segmentation, providing insights into their strengths and limitations. Future work will focus on refining these methods, exploring prototype-based contrastive learning, and validating our approaches on standard benchmarks to ensure broader applicability.

Contents

1 Introduction

Semantic segmentation is one of the core tasks in computer vision, which entails the classification of each pixel in an image into predefined categories. This process is essential for various applications, including medical imaging, autonomous driving, and environmental monitoring. For models to be truly effective in real-world scenarios, they must generalize well across different domains. This means that a model trained on data from one domain should perform accurately when applied to data from a different domain, such as images captured in varying weather conditions or different geographic locations.

1.1 Continual Learning

Continual learning (CLG) is a paradigm in deep learning that addresses the challenge of sequentially learning new information while retaining previously acquired knowledge. This capability is essential for creating models that can adapt to new data over time without requiring complete retraining on the combined old and new datasets. The core idea of CLG is to ensure that models maintain performance on previously learned tasks even as they are trained with new information. This approach is particularly valuable when access to the original training data is restricted or infeasible.
A major challenge in CLG is catastrophic forgetting, where models lose previously learned knowledge when updated with new data [29, 15]. This phenomenon significantly hinders the effective implementation of continual learning strategies. Mitigating catastrophic forgetting is crucial for developing robust CLG systems that can operate effectively in dynamic environments.
Various strategies have been proposed to address this challenge, typically categorized into three main approaches: regularization-based, generation-based, and replay-based methods [48]. Regularization-based methods aim to retain previous knowledge by modifying the neural network architecture or its training process, such as applying weight constraints [21], regularization techniques [24, 49, 32], knowledge distillation [44, 40, 4, 22], and transfer learning. Generation-based methods involve using generative models to produce synthetic data that approximates the original training dataset, thus providing a reference for the model to retain old knowledge [42, 39]. Replay-based methods, on the other hand, maintain a subset of the actual old data and combine it with new data during training to preserve previous knowledge directly [18].
Continual learning can be further divided into domain-incremental and class-incremental learning [48]. Domain-incremental learning involves adapting to new domains with the same set of classes, while class-incremental learning requires the model to learn new classes over time while retaining knowledge of previously learned classes. For more information on these concepts, refer to [48].

1.2 Project Objectives

In this project, we focus on continual learning within the domain-incremental setting of semantic segmentation, particularly in the context of medical image analysis. Medical image segmentation is critical for various clinical applications, such as identifying anatomical structures and surgical instruments during procedures. However, changes in

imaging equipment, such as different cameras, can lead to significant variations in data distribution. Additionally, variations in lighting conditions across different surgery videos can further complicate the segmentation task. These domain shifts require models to adapt effectively without losing performance on previously learned data.
We explore regularization-based, replay-based, and knowledge distillation approaches to address catastrophic forgetting in domain-incremental semantic segmentation. Furthermore, we propose a novel method based on contrastive learning (CL) to enhance the model's ability to distinguish between different features across domains [7].
Our objective is to develop and evaluate methods that enable models to adapt to new domains while preserving their performance on previously learned tasks, ensuring effective and efficient continual learning. By tackling these challenges, we aim to contribute to the advancement of continual learning techniques, ultimately enhancing the robustness and adaptability of semantic segmentation models in dynamic and evolving medical environments.

2 Background

This section presents a foundational overview of the concepts and methodologies underpinning our research. We first discuss semantic segmentation, explaining its importance and the evolution of related techniques. Subsequently, we delve into the Mask2Former model, a state-of-the-art framework adopted as the baseline for our experiments. The concept of universal image segmentation is then explored, encompassing relevant advancements and loss functions for addressing class imbalance. Finally, we examine strategies to mitigate catastrophic forgetting in the context of continual learning, including raw data replay, knowledge distillation, and contrastive learning.

2.1 Semantic Segmentation

Semantic segmentation is a crucial task in the field of computer vision that involves classifying each pixel in an image into a predefined category. Unlike traditional image classification, which assigns a single label to an entire image, semantic segmentation provides a detailed, pixel-level understanding of the scene, enabling more granular and context-aware analysis. This process is fundamental for various applications, from autonomous driving and medical imaging to robotics and agriculture.
In the broader landscape of segmentation tasks, there are several related types. Instance segmentation, for example, not only categorizes each pixel but also differentiates between individual instances of objects within the same category. Panoptic segmentation goes a step further by combining both semantic and instance segmentation to provide a comprehensive framework for detecting and segmenting all objects in a scene. Figure 12 displays the difference of these different types of segmentation methods.
The evolution of semantic segmentation has seen significant milestones. Early methods relies on handcrafted features and traditional machine learning algorithms, which are limited by manual feature extraction and difficulty handling complex scenes. The advent of deep learning, particularly convolutional neural networks (CNNs), revolutionized the field with more accurate and efficient models.

A key development was the Fully Convolutional Network (FCN) [28], which replaced fully connected layers with convolutional layers for pixel-wise classification, enabling end-to-end training and prediction.
Following FCNs, U-Net [37], in 2015, became popular for medical image segmentation due to its encoder-decoder architecture with skip connections, enhancing feature retention and localization accuracy. DeepLab by Chen et al. introduced atrous (or dilated) convolutions and spatial pyramid pooling [5] for better multi-scale context capture. Mask R-CNN [36] extended Faster R-CNN [36] with a branch for segmentation masks, excelling in instance and semantic segmentation.
Our project uses Mask2Former [10], an improved version of MaskFormer [8], as the base model which integrates advanced techniques for handling complex scenes and improving segmentation accuracy. This state-of-the-art model represents the latest advancements in semantic segmentation.

2.2 Mask2Former

Over the past few years, researchers have developed various architectures tailored specifically for instance, semantic, or panoptic segmentation. Instance and panoptic segmentation typically involve generating a set of binary masks with corresponding labels for each object instance, akin to object detection but with binary masks instead of bounding boxes. This approach is known as binary mask classification. In contrast, semantic segmentation was treated as a per-pixel classification problem, where models output a single segmentation map with one label per pixel. Notable models following this paradigm include SegFormer [46] and UPerNet [45].

2.2.1 Universal Image Segmentation

Since around 2020, the field has shifted towards models capable of handling all three tasks - instance, semantic, and panoptic segmentation - using a unified architecture. This shift began with DETR [2], the first model to address panoptic segmentation using a "binary mask classification" paradigm, treating "thing" and "stuff" classes in a unified manner. DETR's key innovation was employing a Transformer [41] decoder to generate a set of binary masks and classes in parallel. This approach was further refined in the MaskFormer paper, demonstrating that the binary mask classification paradigm is also highly effective for semantic segmentation.
Masked-attention Mask Transformer (Mask2Former) builds on this foundation, extending the approach to instance segmentation by enhancing the neural network architecture. This evolution has led to the concept of "universal image segmentation" architectures, capable of solving any image segmentation task. These universal models adopt the "mask classification" paradigm, moving away from the traditional "per-pixel classification" paradigm.

2.2.2 Architecture

The architecture of Mask2Former involves several stages (see Figure 1):

1. **Backbone Processing:** An input image is processed through a backbone network,

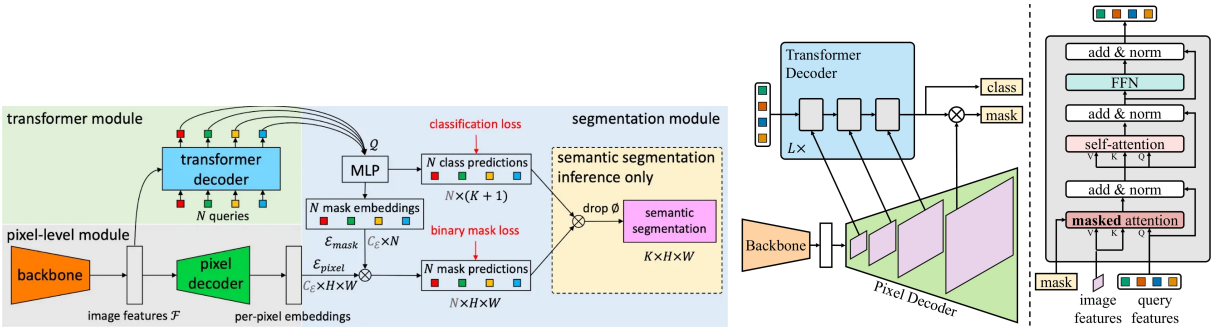


Figure 1: **MaskFormer (left):** A backbone extracts image features $\mathcal{F}$, which are upsampled by a pixel decoder to produce per-pixel embeddings $\mathcal{E}_{\text{pixel}}$. A transformer decoder attends to these features, generating N per-segment embeddings $\mathcal{Q}$, N class predictions, and N mask embeddings $\mathcal{E}_{\text{mask}}$. The model predicts N binary masks via a dot product between $\mathcal{E}_{\text{pixel}}$ and $\mathcal{E}_{\text{mask}}$, followed by sigmoid activation. Final predictions are obtained by combining N binary masks with class predictions through matrix multiplication. **Mask2Former (right):** Mask2Former shares the meta architecture of MaskFormer, consisting of a backbone, pixel decoder, and Transformer decoder. A new Transformer decoder with masked attention is proposed instead of standard cross-attention. For handling small objects, high-resolution features from the pixel decoder are efficiently utilized by feeding one scale of multi-scale features to one Transformer decoder layer at a time. Additionally, the order of self and cross-attention is switched, query features are made learnable, and dropout is removed to enhance computational efficiency

which could be either ResNet [16] or Swin Transformer [27], to generate a set of low-resolution feature maps.

2. **Pixel Decoder Module:** These low-resolution feature maps are then enhanced using a pixel decoder module to obtain high-resolution features.

3. **Transformer Decoder:** Queries, initially abstract feature vectors, are processed by the Transformer decoder through multiple attention layers. Masked attention ensures that each query focuses only on relevant image regions. These refined queries are then used to generate binary masks, identifying specific objects or segments in the image, each with an associated category label. Ultimately, we obtain detailed, labeled segments corresponding to actual objects in the image from the abstract, learnable queries we initially began with.

The pivotal aspect to consider is what sets Mask2Former apart, enabling it to achieve superior performance over MaskFormer (Figure 1, left). Mask2former employs an architecture identical to that of MaskFormer, marked by two primary distinctions: using mask attention instead of cross attention, and the multi-scale high-resolution features with which the model works.

Mask Attention: Mask2Former uses mask attention instead of cross attention to improve instance segmentation. Cross attention, which attends to all pixels including background pixels, can distract the decoder. Mask attention, however, restricts the decoder's focus to the foreground regions, enhancing its ability to concentrate on relevant features

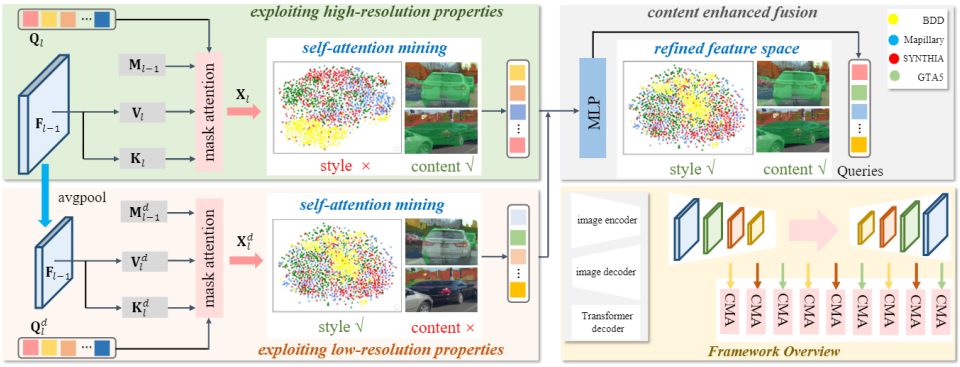


Figure 2: (a) The proposed Content-enhanced Mask Attention (CMA) consists of three key steps, namely, exploiting high-resolution properties (in green), exploiting low-resolution properties (in brown), and content enhanced fusion (in gray). (b) Framework overview (in yellow) of the proposed **C**ontent-enhanced **M**ask Trans**former** (CMFormer) for domain generalized semantic segmentation. The image decoder is directly inherited from the Mask2Former.

and reducing background noise. This also increases efficiency, resulting in shorter training and inference times.

Multi-scale High-resolution Features: To tackle small objects, Mask2Former employs a multi-scale feature representation, capturing details at various resolutions. Each scale is processed by a specific Transformer decoder layer (e.g., 1/32, 1/16, 1/8). This systematic approach enhances the model's capability to manage objects of different sizes effectively.

2.3 Focal Loss

Focal loss [25] is implemented to address the class imbalance in trainings. In our project, both of our datasets have class imbalance therefore, the usage of focal loss could mitigate this issue in our trainings. Focal loss is an improved version of the cross-entropy loss and was originally implemented to mitigate the imbalance between the background and the foreground objects in object detection. This loss function is used in MaskFormer [8] as well. In the below equation [25], p_t is the predicted probability for the instance t, $(1-p_t)^\gamma$ down-weights easy-positive examples in the loss function so that the focus is more on hard-negatives, γ is a hyperparameter that increases the weighting effect when it gets larger and α_t is also a hyperparameter and can be set differently for each class to address the imbalance.

$$\mathcal{L}_{\text{focal}}(p_t) = -\alpha_t(1-p_t)^\gamma \log(p_t) \tag{1}$$

2.4 CMFormer

Content-enhanced **M**ask Trans**former** [1] aims to learn generalized semantic prediction across diverse urban-scene styles in domain-generalized urban-scene semantic segmentation (USSS) setting and we test the performance of this method in our medical semantic segmentation scenario. CMFormer proposes a Content-enhaced Mask Transformer for

domain generalized USSS. The main idea is to enhance the focus of the fundamental component, the mask attention mechanism, in Transformer segmentation models on content information. Their empirical analysis shows that a mask representation effectively captures pixel segments, albeit with reduced robustness to style variations. Conversely, its lower-resolution counterpart exhibits greater ability to accommodate style variations, while being less proficient in representing pixel segments.
Figure 2 depicts the overall architecture of CMFormer. The proposed Content-enhanced Mask Attention (CMA) consists of three key steps, namely, exploiting **high-resolution properties** (in green), exploiting **low-resolution** properties (in brown), and **content enhanced fusion** (in gray).
The fusion of both representations $\tilde{\mathbf{X}}_l$ (high-resolution properties) and $\tilde{\mathbf{X}}_l^d$ (low-resolution properties) is done in a rather simple and straight-forward way. The fused feature $\mathbf{X}_l^{final}$ serves as the final output of the l^{th} Transformer decoder, and it is computed as

$$\mathbf{X}_l^{final} = h_l([\tilde{\mathbf{X}}_l, \tilde{\mathbf{X}}_l^d]), \tag{2}$$

where $[\cdot, \cdot]$ represents the concatenation operation, and $h_l(\cdot)$ refers to a linear layer.

2.5 Knowledge Distillation

In contrast to CMFormer, which mitigates catastrophic forgetting by designing a domain-robust model, knowledge distillation transfers previously learned knowledge to a model provided with new information. This process involves transferring knowledge from a well-trained teacher model to a student model, aligning the student's outputs with the teacher's soft labels to maintain consistency across tasks. Studies have shown that this method effectively stabilizes the learning process and retains critical information from previous tasks, thereby reducing the negative impact of sequential learning on model performance [40, 4, 22, 11].
The Continual Image Segmentation method with incremental Dynamic Query (CiSDQ) [44] is a model-specific knowledge distillation approach tailored towards semantic segmentation using Mask2Former [10]. This method employs dynamic query strategies to adaptively select informative samples from previous tasks, enhancing the retention of important information and mitigating catastrophic forgetting (see Figure 3). By leveraging a dynamic querying mechanism, the model can focus on critical regions of the image that are more likely to be affected by changes in the data distribution across tasks. This technique is especially effective for class-incremental scenarios. Since our focus is on domain-incremental learning, we will concentrate on the knowledge distillation techniques applicable to our setting.
To preserve previously learned representations, the method distills knowledge from either a frozen or slowly updated teacher to the student by applying the following losses:

- Local POD [11, 12] $\mathcal{L}_{pod}$ loss for the backbone and pixel-decoder features.

$$\mathcal{L}(\Theta^t) = \frac{1}{L} \sum_{l=1}^{L} \left\| \Phi(\mathbf{x}_l^t) - \Phi(\mathbf{x}_l^{t-1}) \right\|^2, \tag{3}$$

  where t is the current task (student), $t-1$ is the previous task (teacher), x_l is the activation from the l–th layer, and Φ is a mapping function.

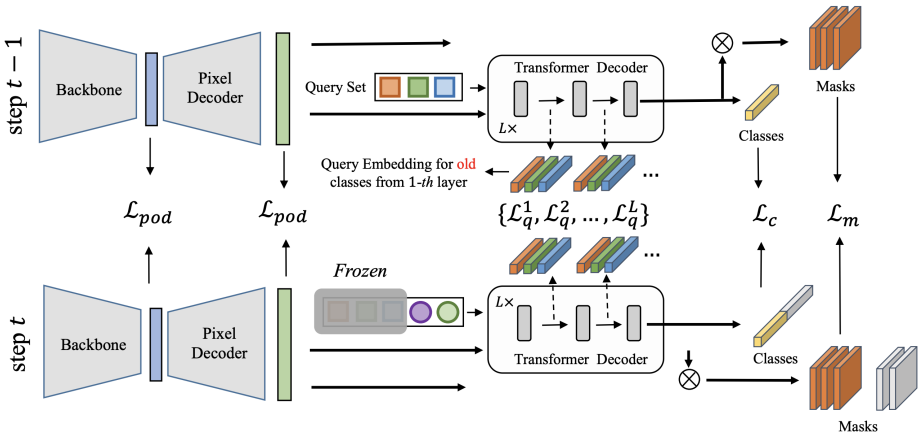


Figure 3: **Query Guided Knowledge Distillation** proposed in CiSDQ [44]. It consists of three parts: 1) Local POD loss [11, 12] $\mathcal{L}_{pod}$ for features from the backbone and the pixel-decoder. 2) L_2 loss for the learned queries $\mathcal{L}_q$. 3) Loss for the network prediction $\mathcal{L}_c$ and $\mathcal{L}_m$.

- L_2 loss for the queries $\mathcal{L}_q$ after each transformer-decoder layer.

$$\mathcal{L}_q(\Theta^t) = \frac{1}{M}\frac{1}{L}\sum_{j=0}^{M}\sum_{l=1}^{L}\left\|\mathbf{q}_{j,l}^t - \mathbf{q}_{j,l}^{t-1}\right\|^2, \tag{4}$$

  where M is the number of all query embeddings and L is the number of transformer-decoder layers.

- Kullback-Leibler (KL) divergence for the class predictions $\mathcal{L}_c$.

$$\mathcal{L}_c(\Theta^t) = \sum_{i=0}^{t-1} c_i^{t-1}\log\frac{c_i^{t-1}}{c_i^t}, \tag{5}$$

  where c_i is the i-th class distribution prediction associated with the query set $Q \in \{Q_0, Q_1, \ldots, Q_M\}$ and M is the number of learnable queries.

- Dice [31] and cross-entropy loss [9] for the masks $\mathcal{L}_m$.

$$\mathcal{L}_m(\Theta^t) = \lambda_c\frac{1}{M}\sum_{j=0}^{M}\mathcal{L}_{ce}(m_j^t, m_j^{t-1}) + \lambda_d\frac{1}{M}\sum_{j=0}^{M}\mathcal{L}_{dice}(m_j^t, m_j^{t-1}), \tag{6}$$

  where m is the predicted mask. λ_c and λ_d are two weight parameters, which are similar to Mask2Former [10].

2.6 Raw Data Replay Methods

Most benchmarks for continual semantic segmentation typically consist of sequential tasks where all classes are present. Consequently, knowledge distillation is widely adopted in this context. However, this method tends to underperform on benchmarks where the sets

of classes between sequential tasks are disjoint, i.e., each task involves learning to segment new classes. A straightforward solution to address this issue is to replay classes from previous tasks. This approach has proven effective in both class-incremental and domain-incremental scenarios [19]. Nonetheless, due to the constraints of a limited memory buffer, selecting the "right" samples for replay is crucial for the efficacy of this method. The study by [18] explores various approaches to sample instances from previous tasks to minimize forgetting. Furthermore, the findings of [18] demonstrate that replay methods outperform knowledge distillation in domain-incremental scenarios, which aligns with the focus of our research project. Therefore, we will introduce only those methods that perform well in this setting.

2.6.1 Loss-Based Selection

This method selects samples based on their loss values – either **maximum, minimum, or mean**. The underlying idea for selecting samples with the minimum loss is that the neural network adapts well to frequent scenarios in the dataset, resulting in minimum loss values for these samples. Conversely, selecting samples with the maximum loss, although less representative, may still be valuable for replay due to their high information content. Additionally, mean loss selection can be used to avoid selecting both trivial minimum-loss samples and maximum-loss outliers, providing a balanced approach.

2.6.2 Representation-Based Sample Selection

Representation-based sample selection (RSS) [18] aims to select buffer samples that approximate the learned representations of classes from previous tasks. This method utilizes the latent space representations produced by the image encoder for each sample, projecting them into a lower-dimensional space using UMAP [30]. The projected activations are then grouped into M clusters using k-means clustering, where the number of clusters corresponds to the number of classes. The samples closest to these cluster centers are subsequently selected for replay.
Since we use Mask2Former [10] as our base model, we perform RSS using the backbone features, which are extracted by a Swin Transformer [27] (see Figure 1).

2.7 Latent Replay

The "Latent Replay for Real-Time Continual Learning" [33] proposes replaying latent representations rather than raw data, as described in the previous section. The paper suggests storing latent vectors instead of raw inputs for the replay trainings in continual learning. For the same memory space, more latent vectors can be stored due to their smaller sizes compared to the raw inputs. Testing this theory in our project can be interesting since the model could benefit from the increased number of replayed samples. Additionally, this method emphasizes that the low-level layers, which are closer to the raw input, are significant for the generic feature extraction and they remain stable after being trained on a large dataset. However, high-level layers should be extensively fine-tuned for each new task. The training of the low-level layers must be slowed down to prevent the vectors from becoming irrelevant to the trained model (aging effect). During the replay training, the replayed latent vectors are merged with the new latent vectors in the current

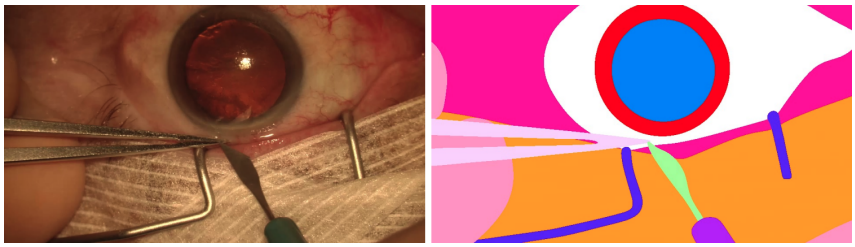


Figure 4: Example image frame (left) and semantic segmentation labels (right) from the Cataract dataset for Image Segmentation presented in CaDIS [13]. (Colormap: Pupil, Iris, Cornea, Skin, Surgical tape, Eye retractors, Hand, Bonn Forceps, Secondary Knife and Secondary Knife Handle)

batch at the replay layer, where the latent vectors were previously saved, e.g. bottleneck layer.

2.8 Contrastive Learning

Contrastive learning aims to learn good representations of the anchor instances in the latent space by pulling positive instances closer to the anchor while pushing negative instances away. It is used in various deep learning applications as it significantly enhances the performance of the model in distinguishing different classes [6, 20, 43]. Therefore, in our project we experiment with contrastive loss in semantic segmentation setting where the classification occurs pixel-wise. There have been self-supervised [6] and supervised [20] implementations of contrastive learning. For this project, a supervised contrastive loss (CL) is required. The pixel-wise supervised contrastive loss from the "Exploring Cross-Image Pixel Contrast for Semantic Segmentation" [43] is used:

$$\mathcal{L}_i^{NCE} = \frac{1}{|\mathcal{P}_i|} \sum_{i^+ \in \mathcal{P}_i} -\log \frac{\exp(i \cdot i^+/\tau)}{\exp(i \cdot i^+/\tau) + \sum_{i^- \in \mathcal{N}_i} \exp(i \cdot i^-/\tau)} \tag{7}$$

where i is the feature vector for the anchor pixel, $\mathcal{P}_i$ is the set of positive samples, $\mathcal{N}_i$ is the set of negative samples for the anchor pixel i and τ is the temperature variable that scales the similarities between the pairs. An additional 1x1 convolutional projection head is appended at the end of the encoder network to project the high-dimensional features into 256-dimensional vectors. These vectors are then l_2-normalized before being used in the loss calculation. The anchor pixels in the loss are sampled from the same image, the other images in the batch, and the per-class region memory bank, where the average pixel embeddings from the latest batch are stored. Furthermore, hard-anchor sampling is employed to select positive and negative pixels. Each unique class in an image from the batch becomes an anchor class. Instead of using all the pixels in the images, hard-negative and easy-positive pixels are collected and then randomly sampled for each anchor class. Hard-negative pixels have different ground truth labels than the anchor class, but the model incorrectly predicts their labels as the anchor class. The easy-positive pixels have the same ground truth classes as the anchor class, and the model also correctly predicts them. It is aimed to sample an approximately equal number of easy-positive pixels and hard-negative pixels when there are enough pixels for each of category.

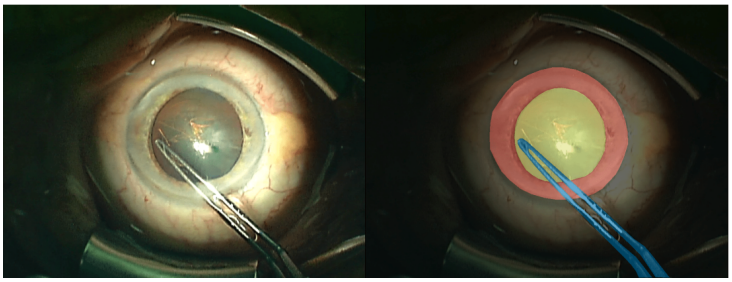


Figure 5: Example image frame (left) and semantic segmentation labels (right) from the Cataract–1K dataset [**Cataract-1K**]. (Colormap: Katena Forceps, Iris, Pupil

2.9 Prototypes

In the "Decoupled Semantic Prototypes Enable Learning from Diverse Annotation Types for Semi-weakly Segmentation in Expert-Driven Domains" [35], trainable vectors that are known as prototypes are used as anchors in the decoupled contrastive loss (DCL) [47]. Prototypes are 256-dimensional class-specific representations. For each class, more than one prototype is created to capture the variations within the class. They represent intra-class information and inter-class dissimilarities effectively and hence, they have the potential to improve the performance of a standard contrastive loss even further. Therefore, we experiment with prototypes by using them as pre-trained anchors in our Cataract-1K trainings with CL.

3 Datasets

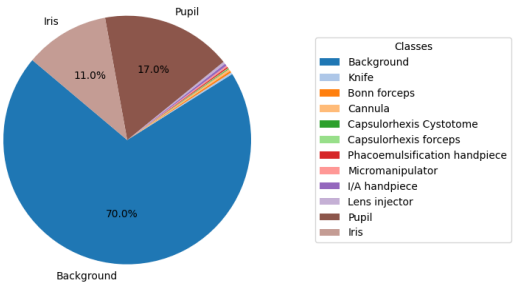
In this section, we introduce the datasets used in our experiments – Cataract dataset for Image Segmentation (CaDIS) [13] and Cataract–1K [14]. We describe the domain-incremental scenario with class mappings and provide exploratory data analysis (EDA) on class distribution and errors.

3.1 CaDIS

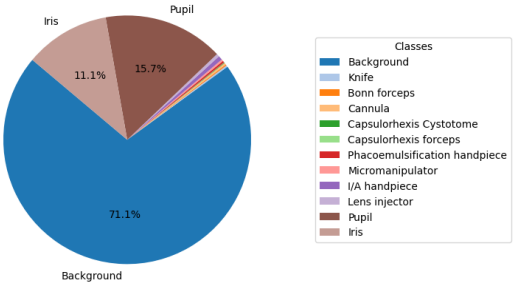
The CaDIS dataset is a comprehensive dataset designed for semantic segmentation tasks in the context of cataract surgery. It consists of 4,670 high-resolution surgical images, providing pixel-level annotations for various anatomical structures and surgical instruments. An example image with its corresponding semantic map is illustrated in Figure 4.

3.2 Cataract–1K

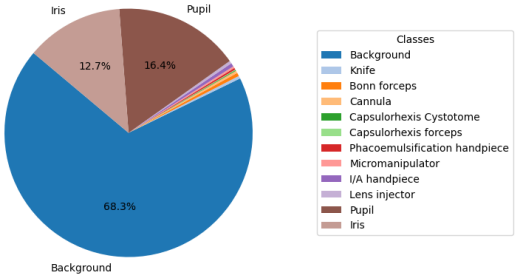
The Cataract–1K dataset is a large-scale collection of annotated images specifically curated for the task of cataract surgery analysis. This dataset includes 2,256 annotated images captured during cataract surgeries, with detailed labels for different surgical phases, instruments, and anatomical structures. An example image with its corresponding semantic map is illustrated in Figure 5.



(a) CaDIS Train Distribution



(b) CaDIS Validation Distribution



(c) CaDIS Test Distribution

Figure 6: Class Distributions for CaDIS Dataset

3.3 Domain-Incremental Scenario with Class Mappings

In this work, we explore catastrophic forgetting within a domain-incremental scenario, where class consistency is maintained while the data domain shifts. Directly using the datasets without preprocessing is not feasible. Consequently, we initially define a set of common classes across the two domains, termed *Common Classes*.
The Common Classes include two categories: anatomical structures and surgical instruments. The anatomical category comprises *Iris* and *Pupil*. The instruments category includes *Knife*, *Bonn Forceps*, *Cannula*, *Capsulorhexis Cystotome*, *Capsulorhexis Forceps*, *Phacoemulsification Handpiece*, *Micromanipulator*, *Lens Injector*, and *I/A Handpiece*.
We map the classes from both datasets to Common Classes. Any classes that are not mapped to the Common Classes are considered as *background*. The detailed mappings from CaDIS and Cataract–1K classes to Common Classes are presented in Table 1.

3.4 Explorative Data Analysis

In this section, we analyze the class distributions and annotation errors in the CaDIS and Cataract–1K datasets.

3.4.1 Class Distribution

From the distributions illustrated in Figures 6 and 7, the following observations can be made for both datasets:

- The *Background* class overwhelmingly dominates the distribution across all splits, accounting for approximately 70% in CaDIS and 84% in Cataract-1K.

Common Class	CaDIS Classes	Cataract–1K Classes
Knife	Primary Knife Secondary Knife	Incision Knife Slit Knife
Bonn forceps	Bonn Forceps	Katena Forceps
Cannula	Hydrodissection Cannula Viscoelastic Cannula Rycroft Cannula Charleux Cannula	Gauge
Capsulorhexis Cystotome	Capsulorhexis Cystotome	Capsulorhexis Cystotome
Capsulorhexis Forceps	Capsulorhexis Forceps	Capsulorhexis Forceps
Phacoemulsification Handpiece	Phacoemulsifier Handpiece	Phacoemulsification Tip
Micromanipulator	Micromanipulator	Spatula
I/A handpiece	I/A Handpiece	I/A Device
Lens injector	Lens Injector	Lens Injector
Pupil	Pupil	Pupil
Iris	Iris	Iris
Background	Rest	Rest

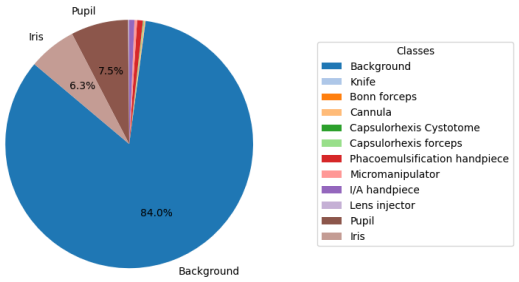
Table 1: Mappings from CaDIS and Cataract–1K classes to *Common Classes*. The rest of the classes that are not mentioned in the mappings are cosidered as background.

- The *Pupil* class is the second most frequent, constituting about 16% of the data in CaDIS and 7% in Cataract-1K.
- The *Iris* class follows, representing around 11% of the data in CaDIS and 6% in Cataract-1K.
- The remaining classes collectively make up approximately 3% of the distribution.

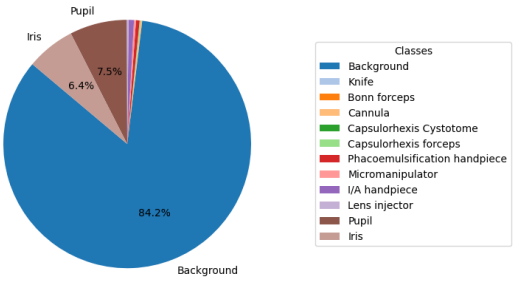
These observations indicate a significant class imbalance in both datasets, which must be carefully addressed during model training.

3.4.2 Error Analysis

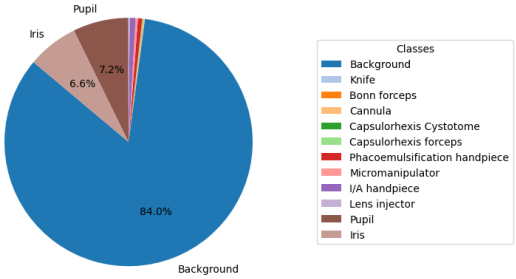
Annotation accuracy is critical for model training in image segmentation. However, several sources of error have been identified in the annotations. According to the authors of the CaDIS dataset [13], blurriness from instrument or patient motion can lead to unclear boundaries, and specular reflections can obscure precise delineation, particularly at instrument tips within anatomical structures. Despite efforts to maintain accurate boundaries, these issues persist.
We conducted a qualitative inspection of a subset of images and confirmed the presence of annotation errors. For example, Figure 8 illustrates an example of such an error due to blurriness.
Despite the presence of annotation errors, it is common practice to train models on these datasets without excluding the erroneous samples [34]. Therefore, we have also chosen not to remove samples with errors in our training process.



(a) Cataract Train Distribution



(b) Cataract Validation Distribution



(c) Cataract Test Distribution

Figure 7: Class Distributions for Cataract-1K Dataset

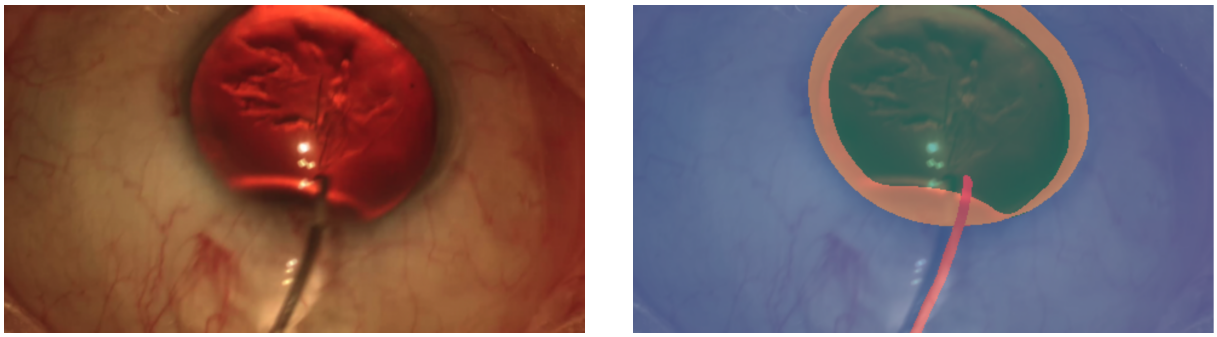
4 Methods

This section outlines the methodologies and experimental setups employed in our study. We detail the training configurations, augmentation techniques, raw data replay strategies, knowledge distillation approaches, latent replay methods, focal loss application, and contrastive learning procedures.

4.1 Training Setup

The training setup for our project involves a carefully selected combination of model, environment, and preprocessing steps to ensure optimal performance and efficiency. Here are the key components of our training setup:

- **Model:** We use Mask2Former as the baseline model, leveraging the Huggingface implementation for its robust features and ease of use.
- **Backbone:** The model employs a tiny Swin Transformer pretrained on ImageNet, which enhances feature extraction and overall model performance.
- **Preprocessing:** Input images are resized to 270x480 pixels to maintain consistency, masks with nearest-neighbor interpolation and images with bilinear interpolation. Normalization is performed according to the training set specifications at each step, ensuring the data is in a suitable format for training.
- **Training Parameters:** The model is trained for 200 epochs. Early stopping is implemented with a patience of 15 epochs to prevent overfitting.



(a) Image (b) Ground Truth Segmentation Map

Figure 8: An example image that contains an annotation error from the CaDIS dataset [13]. The instrument Capsulorhexis Cystotome is annotated slightly to the right of its actual position.

This setup ensures a structured and efficient training process, leveraging advanced tools and techniques to achieve the best possible performance with Mask2Former.

4.2 Augmentations

In our project, we focus on various augmentations to improve model generalization and reduce the domain generalization gap. We explore several augmentation techniques and evaluate their effectiveness in enhancing the model's performance across different domains. Here are the key augmentations we employed:

- **Random Cropping**: This technique involves randomly cropping a quarter-sized portion from the resized input.
- **Color Jittering**: This augmentation randomly changes the brightness, contrast, saturation, and hue of the input image. It simulates different lighting conditions and color variations, aiding the model in learning color-invariant features.
- **ColorAugSSDTransform**: This augmentation, inspired by the "SSD: Single Shot MultiBox Detector paper" [26], includes a series of color transformations applied to the image. It involves:
  - Brightness Adjustment: Randomly altering the image brightness within a specified range.
  - Contrast Adjustment: Modifying the image contrast by randomly scaling the intensity.
  - Saturation Adjustment: Changing the image saturation by converting it to HSV color space and adjusting the saturation channel.
  - Hue Adjustment: Altering the hue of the image by converting it to HSV color space and shifting the hue channel.

  This sequence of transformations is applied in random order, making the model robust to various color variations.

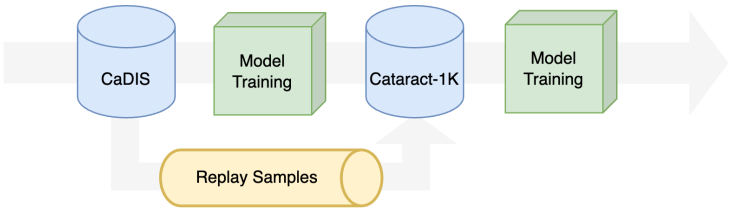


Figure 9: Replay Training Pipeline

- **Combination of Augmentations**: We also experiment with applying combinations of the aforementioned augmentations. Each combination was applied with probabilities $p = 1.0$ or $p = 0.5$, allowing us to assess the impact of consistent versus occasional augmentation.

4.3 Raw Data Replay

One of the methods we employ is raw data replay, as previously introduced in Section 4.3. Based on prior research demonstrating their effectiveness in domain-incremental scenarios [18], we selected methods utilizing minimum, mean, and maximum loss selection, as well as representation-based sample selection. The training pipeline is depicted in Figure 9.

4.3.1 Replay Strategies

The initial step involves training a base model on the CaDIS dataset. Subsequently, we apply one of the replay strategies to select samples from CaDIS to be replayed during the training on the Cataract-1K dataset. Two primary replay strategies are employed:

1. **Merged Replay:** Selected samples are merged with the new dataset. This method has the advantage of faster training times, as the increase in training samples is minimal. However, the limitation is that the replayed samples may not appear frequently enough to effectively mitigate forgetting.

2. **Balanced Batch Replay:** Each training batch is composed of 50% new dataset samples and 50% memory buffer samples. This approach ensures more frequent replay of previous samples, addressing the limitations of the merged replay strategy. However, it doubles the training data size, resulting in longer training times and creates a potential risk of overfitting on the limited replayed samples.

4.3.2 Memory Buffer

Previous studies have experimented with various memory buffer sizes, typically 32, 64, or 128 MB [18]. Larger memory buffers allow for the storage of more samples. In this study, we assume a constrained memory environment and utilize a 32 MB memory buffer, which accommodates only 21 images from the CaDIS dataset.

4.4 Knowledge Distillation

We employ knowledge distillation to mitigate forgetting, following a similar approach as described in Section 2.5. The method proposed by CiSDQ [44] is specifically tailored

for the Mask2Former architecture [10] and is designed for class-incremental learning. To address the challenge of learning new classes while retaining previously learned ones, their method involves freezing the learnable queries trained in the previous task during the training on the current task (see Figure 3).
Since our pipeline does not introduce new classes, we do not adopt this freezing mechanism. Instead, we utilize the losses defined in Equations 3, 4, 5, and 6 to distill knowledge from the model trained on CaDIS to the model being trained on Cataract-1K. We define two different approaches for this training:

1. **Frozen Teacher**: In this approach, the teacher model is frozen during distillation and is not updated.

2. **Updated Teacher**: In a manner similar to self-distillation with no labels (DINO) [3], we update the teacher model using an exponentially moving average of the student parameters:
$$\theta_t \leftarrow \lambda\theta_t + (1-\lambda)\theta_s, \tag{8}$$
where θ_t and θ_s represent the parameters of the teacher and student models, respectively, and λ is a decay hyperparameter set to 0.999.

4.5 Latent Replay

As in the 32MB raw data replays in Figure 9, we replay latent vectors of 21 CaDIS samples at the bottleneck layer. The number of latent vectors that fit into 32MB is not calculated using the bottleneck vector size because, in Mask2Former, the pixel decoder also uses some additional intermediate encoder features along with the bottleneck vector. As a result, unfortunately, latent replay with Mask2Former requires more storage than raw data replays. To compare the performance of the latent replay training against other replay methods, at the end of the CaDIS training, 21 images are randomly sampled again, and their latent vectors are saved. In each batch iteration in replay training, 8 latent vectors from the CaDIS replay set are randomly sampled and merged with 8 new Cataract-1K latent vectors in the current batch at the bottleneck layer of the model. The batch size becomes 16 after the bottleneck layer. Lastly, the encoder of the model is completely frozen during the replay training.

4.6 Focal Loss

The effect of focal loss on class imbalance is tested by adding it to the CaDIS training. The focal loss is multiplied by one and added to the default losses of the Mask2Former. All pixel predictions are used in the focal loss.

4.7 Contrastive Learning

The pixel-level supervised contrastive loss and the hard-anchor sampling [43] are adapted for the purpose of this project. After that, the effect of the weighted contrastive loss and prototypes as anchors are observed.

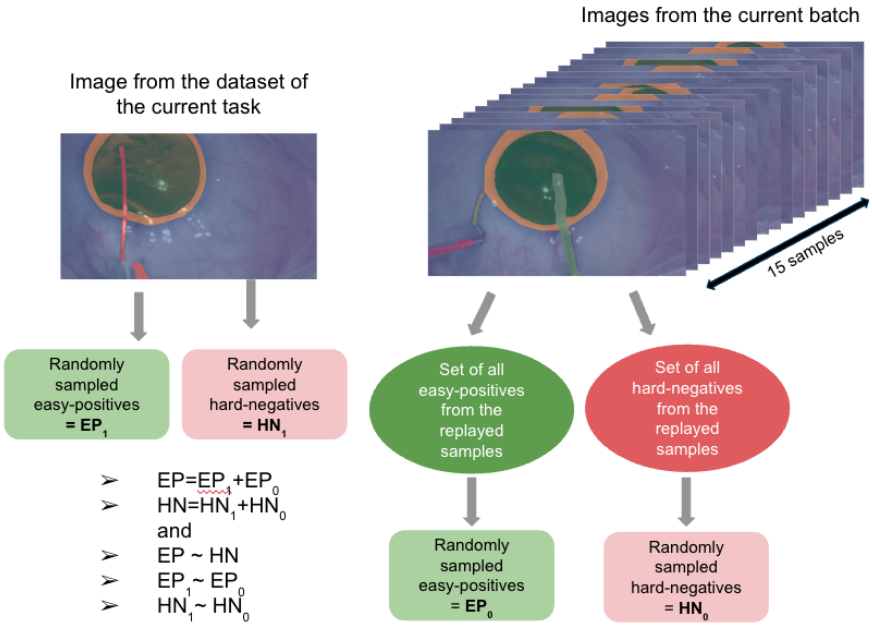


Figure 10: Pixel sampling pipeline in contrastive learning without replayed samples in the batch.

4.7.1 Contrastive Learning with Mask2Former

Contrastive loss requires pixel-level features, which is the reason why we utilize the pixel decoder features from the Mask2Former architecture (see Figure 1). The queries in the transformer are trained to ask class-specific questions to understand the class semantics, but they do not retain image-specific information. The rationale is that through contrastive learning, the transformer block queries should generalize the class semantics and remain relatively stable, even with domain changes. In contrast, the pixel decoder features contain image-level information that varies with each image or domain. Thus, the pixel decoder features answer class-specific questions posed by the queries in the transformer block (see Figure 1). The four pixel decoder features are upscaled to the size of the input image, and their pixel-wise average provides a single per-pixel feature matrix.

4.7.2 Sampling Method for Contrastive Learning

In the original implementation of pixel-wise contrastive loss [43], each unique class in each image serves as an anchor class. In our approach, the definition of "anchor class" varies depending on the type of training.
During standard training sessions without continual learning methods (e.g., CaDIS training), all images in the batch contribute to the calculation of anchor classes, as illustrated in Figure 10. Conversely, in the subsequent continual learning tasks with replay strategies which are proven to be very effective at mitigating forgetting (e.g., Cataract-1K training with replay), only the images from the current task are used to determine the anchor classes, as shown in Figure 11.
Moreover, in standard training sessions, for each unique class in an image, easy-positive and hard-negative pixels are sampled equally from within the image and from the com-

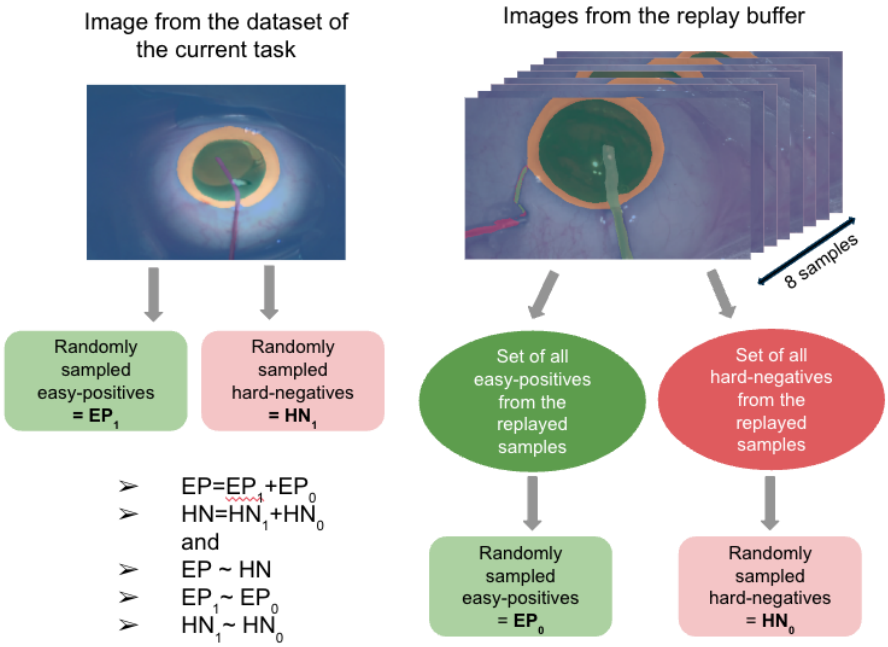


Figure 11: Pixel sampling pipeline in contrastive learning with replayed samples in the batch.

bined pool of all other images in the batch. In contrast, during continual learning tasks, for each unique class in a given image from the current task, pixels are sampled equally from within that image and from the pool of all images from previous tasks included in the batch. The total number of easy-positive and hard-negative pixels is kept equal, as is the distribution of sampled pixels from the current image and the relevant other images in the batch. If there are insufficient pixels in any category, additional pixels are sampled from the complementary set to compensate for the shortfall.

4.7.3 Weighted Contrastive Loss

In order to address the class imbalance in the datasets, the weighted contrastive loss is implemented (see Figures 6 and 7). Class distributions, p_i, are calculated for the training dataset of the current task prior to training. We denote the *Common Classes* shown in Table 1 as $\mathcal{C}$ and calculate the weight w_c assigned to each class $c \in \mathcal{C}$ as:

$$w_c = \frac{1 - p_i}{\sum_{j \in \mathcal{C}} (1 - p_j)}, \tag{9}$$

The calculated weights are used in our pixel-wise contrastive loss.

4.7.4 Contrastive Learning with Prototypes

In the original pixel-wise contrastive loss implementation [43], all sampled pixels act as anchors. To enhance our CL loss, we experiment with using prototype vectors as anchors. Following the approach described in [35], we use five 256-dimensional prototypes per class.

At the start of the first task training with CL, prototype vectors are randomly initialized and trained alongside the model with the same learning rate, 10^{-4}. By the end of this training, these prototypes have effectively learned class semantics and can be utilized as pre-trained anchors in the subsequent second task training. During the following task trainings, the prototypes are fine-tuned with the same learning rate as the Swin transformer backbone, 10^{-5}. The training of prototypes is deliberately slowed down using a lower learning rate to prevent them from changing significantly. Therefore, the class semantics learned from the first dataset can be preserved more. The usage of prototypes as transferred anchors offers a significant potential in learning and transferring domain-invariant class information in a more effective way than standard CL approaches.

5 Evaluation

In this section, we evaluate our proposed methods. We compare various replay strategies, knowledge distillation techniques, and the use of contrastive learning with and without augmentation. The evaluation is conducted using standard metrics and tools to ensure the reliability and reproducibility of our results.

5.1 Tools

For the project development, the following tools are used: Hugging Face for transformer models, Weights and Biases to log the results and save the best performing models, Paperspace to run the experiments. NVIDIA RTX A6000 is used for our contrastive learning experiments and NVIDIA Quadro P6000 for other experiments.

5.2 Evaluation Metrics

After training, we evaluate the model on the test set of the current dataset task using the mean intersection-over-union (mIoU). Following established practices in the study of catastrophic forgetting [17], we denote the mIoU of the model trained on all tasks up to p and evaluated on task q as $mIoU_{p,q}$.
We utilize the *average learning accuracy* metric to assess the model's performance across the two domains (CaDIS and Cataract-1K):

$$\text{average learning acc.} = \frac{\text{mIoU}_{0,0} + \text{mIoU}_{1,1}}{2} \tag{10}$$

Additionally, we employ the *forgetting* score to quantify the decrease in performance on the first task after training on the second task:

$$\text{forgetting} = \text{mIoU}_{0,0} - \text{mIoU}_{1,0} \tag{11}$$

It is important to note that during the training on different tasks, we use the mIoU on the validation sets to evaluate the model's performance on the current task. The forgetting metric is not considered when selecting the best model from the current training loop.

5.2.1 Notations

We introduce the following notations used in this study. The training is conducted first on the CaDIS dataset, referred to as the 0-th task, followed by training on the Cataract-1K dataset, referred to as the 1-st task. We define the following mIoU scores of interest:

- $mIoU_{0,0}$: The mIoU score of a model trained on CaDIS and evaluated on CaDIS.
- $mIoU_{0,1}$: The mIoU score of a model trained on CaDIS and evaluated on Cataract-1K.
- $mIoU_{1,0}$: The mIoU score of a model trained on both CaDIS and Cataract-1K, and evaluated on CaDIS.
- $mIoU_{1,1}$: The mIoU score of a model trained on both CaDIS and Cataract-1K, and evaluated on Cataract-1K.

5.3 Experiments

In this section, we present a comprehensive evaluation of our proposed methods. We begin by establishing baseline performance metrics to assess the impact of catastrophic forgetting during training transitions. Next, we explore the effectiveness of various augmentation strategies in enhancing domain generalization. We then evaluate the CMFormer method and several replay strategies to mitigate forgetting. Additionally, we compare different knowledge distillation techniques and examine the role of contrastive learning both in isolation and combined with other methods. Finally, we investigate the use of prototype vectors as anchors in contrastive learning.

5.3.1 Baseline Trainings

Table 2 summarizes the evaluation of two baseline methods for assessing forgetting during training transitions:

- **Naive Forgetting**: Training initially on CaDIS followed by training on Cataract-1K dataset resulted in a forgetting score of 14.47.
- **Replay All**: Replaying the entire CaDIS dataset during training on Cataract-1K leads forgetting score of **6.79**.

We can think of these numbers as upper and lower bounds for the forgetting score. Additionally, when trained solely on the Cataract-1K dataset, the model achieved a mIoU score of 77.84. In the following sections we try to improve the baseline results by applying various methods.

Method	**mIoU$_{0,0}$**	**mIoU$_{1,1}$**	**mIoU$_{1,0}$**	**Forgetting**	**Avg. Learning Accuracy**
Naive Forgetting	75	79.60	60.17	14.47	77.12
Replay All	75	75.24	67.52	**6.79**	74.78

Table 2: Performance results of baseline methods.

5.3.2 Augmentation Trainings

Based on the experiments summarized in Table 3, we evaluate various augmentation strategies to enhance domain generalization. Among these, color jittering augmentation yields the highest mIoU score of **28.24**. Consequently, we adopt color jittering augmentation for subsequent augmentation trainings due to its superior domain generalization performance. Lastly, it can be seen that focal loss does not give any improvements so we didn't continue using this loss.

Method	**mIoU$_{0,0}$**	**mIoU$_{0,1}$**
No Augmentation	74.64	27.17
No Augmentation + Focal Loss	73.64	22.95
Color Jitter (p = 0.5)	72.23	**28.24**
Random Cropping	69.20	22.82
Color Jitter + Random Cropping	71.78	22.43
ColorAugSSDTransform (p = 0.5)	70.38	28.05
ColorAugSSDTransform (p = 1.0)	75.15	25.21

Table 3: Performance results of domain generalization experiments.

5.3.3 CMFormer Trainings

We evaluate the effectiveness of the CMFormer method in reducing domain gaps and present the results in Table 4, where "All" refers to the following augmentations: ResizeShortestEdge, RandomCrop, ColorAugSSD, and RandomFlip all together. These results do not surpass the best augmentation method, ColorJitter, that was simply aplied on top of Mask2Former model as described in the previous section.

Method	**mIoU$_{0,0}$**	**mIoU$_{0,1}$**
No Augmentation	72.33	23.35
Random Crop	71.25	18.13
ColorAugSSDTransform	**73.85**	**27.69**
ColorAugSSDTransform + Random Crop	68.72	25.55
All	71.70	16.94

Table 4: Performance results of CMFormer experiments.

We also conduct a forgetting experiment with the best-performing augmentation, ColorAugSSD, and obtain a forgetting score of 15.62, which again does not compare well with other experiments and eventually decide to continue the subsequent experiments without CMFormer.

5.3.4 Replay Methods

We conduct multiple experiments using all the replay strategies introduced in Sections 4.3 and 4.5, and present the results in Table 5. The backbone parameters were initialized

using a pre-trained model on ImageNet [38], which is why we also include experiments with a frozen backbone.
Our findings suggest that balanced batch replay generally outperforms merged batch replay, as observed in the results for random replay. Additionally, using a small learning rate for the backbone yields better results than freezing it. Furthermore, mean loss replay appears to be the most promising approach to mitigate forgetting while simultaneously learning new knowledge.

Method	**mIoU$_{0,0}$**	**mIoU$_{1,1}$**	**mIoU$_{1,0}$**	**Forgetting**	**Avg. Learning Accuracy**
Naive Forgetting	75	79.6	60.17	14.47	77.12
Random Replay (*Merged*)	75	79.73	60.16	13.4	76.64
Random Replay† (*Merged*)	75	<u>80.58</u>	58.89	15.75	<u>77.61</u>
Random Replay (*Balanced Batch*)	75	79.89	63.68	10.96	77.27
Random Replay† (*Balanced Batch*)	75	78.65	63.5	11.15	76.65
Min. Loss Replay (*Balanced Batch*)	75	77.36	59.64	15	76
Max. Loss Replay (*Balanced Batch*)	75	79.27	57.48	17.16	76.95
RSS Replay (*Balanced Batch*)	75	80.05	59.33	15.31	77.34
Latent Replay† (*Balanced Batch*)	75	80.15	<u>64.48</u>	<u>10.16</u>	77.34
Mean Loss Replay (*Balanced Batch*)	75	**82.59**	**64.63**	**10.01**	**78.61**

Table 5: Performance results of replay-based methods. Methods marked with † are trained with a frozen backbone. The results for random replay methods are averaged over 3 experiments.

We hypothesize that minimum loss selection chooses samples that are relatively trivial, leading to model overfitting. On the other hand, maximum loss selection picks samples that are outliers and likely contain annotation errors, as discussed in Section 3.4.2. This results in higher forgetting compared to naive sequential training. Consequently, we decide to utilize mean loss replay with the balanced batch strategy for the remainder of our experiments in this study.

5.3.5 Knowledge distillation

We compare the knowledge distillation techniques described in Section 4.4 with naive forgetting and mean loss replay in Table 6. Additionally, we enhance the standard distillation

approach by incorporating mean loss replay. The results indicate that the integration of mean loss replay with knowledge distillation does not reduce forgetting compared to mean loss replay alone. In one of the experiments, this approach even results in higher forgetting compared to naive sequential training, which we consider the baseline for the worst-case scenario.

Method	**mIoU$_{0,0}$**	**mIoU$_{1,1}$**	**mIoU$_{1,0}$**	**Forgetting**	**Avg. Learning Accuracy**
Naive Forgetting	75	79.6	60.17	14.47	77.12
Mean Loss Replay (*Balanced Batch*)	75	**82.59**	**64.63**	**10.01**	**78.61**
Knowledge Distil. (*Frozen Teacher*)	75	75.92	63.77	10.87	75.28
Knowledge Distil. (*Updated Teacher*) (*Mean Loss Replay*) (*Balanced Batch*)	75	<u>81.51</u>	63.53	11.11	<u>78.07</u>
Knowledge Distil.* (*Updated Teacher*) (*Mean Loss Replay*) (*Balanced Batch*)	75	76.95	53.8	20.84	75.79
Knowledge Distil. (*Updated Teacher*)	75	77.57	<u>63.85</u>	<u>10.79</u>	76.11

Table 6: Performance results of knowledge distillation methods. Methods marked with * include hyperparameter optimization.

These findings suggest that the current knowledge distillation methods, even when combined with mean loss replay, are insufficient for mitigating catastrophic forgetting. Therefore, we focus on exploring alternative strategies to improve the model's retention of previously learned knowledge while learning new tasks.

5.3.6 Contrastive Learning only on Cataract-1K

Initially, the performance of the contrastive loss (CL) is tested on mean loss training, which performs the best among the replay methods (see Table 5), using the standard CaDIS training weights (without CL or any other modifications).
This approach is beneficial as it does not require additional training on CaDIS with CL. A pre-trained model on the first dataset is sufficient to enhance metrics on new dataset trainings with improvements like CL. The results in Table 7 support this hypothesis. A low forgetting score of 9.38% is obtained without compromising too much on the Cataract-1K test set predictions. On the other hand, the weighted-CL training increases the $mIoU_{1,1}$ as expected; however, the forgetting score also increases. This shows that whenever the model becomes too confident about the new dataset, forgetting of the old dataset increases. This suggests that there is a trade-off between the old and new information.

Method	$\mathbf{mIoU_{0,0}}$	$\mathbf{mIoU_{1,1}}$	$\mathbf{mIoU_{1,0}}$	Forgetting	Avg. Learning Accuracy
Naive Forgetting	75	79.6	60.17	14.47	77.12
Replay All	75	75.24	**67.52**	**6.79**	74.48
Mean Loss Replay with CL	75	78.65	65.26	9.38	76.65
Mean Loss Replay with CL-W	75	**80.33**	64.53	10.11	**77.48**

Table 7: Performance comparison of mean loss replay with contrastive learning (CL) and mean loss replay with weighted contrastive learning (CL-W) against naive forgetting and replay all.

5.3.7 Contrastive Learning on Both Datasets

Method	$\mathbf{mIoU_{0,0}}$	$\mathbf{mIoU_{0,1}}$
CL	71.72	25.18
CL with Augmentation	**75.05**	**26.72**

Table 8: Performance of Mask2Former on CaDIS dataset using contrastive learning with and without color jitter augmentation.

After the successful results in Section 5.3.6, the performance of the model is also tested when CL is incorporated into the CaDIS training. Having identified color jitter as the most effective augmentation method in the augmentation experiments, color jitter is employed in the subsequent CL experiments.

Method	$\mathbf{mIoU_{0,0}}$	$\mathbf{mIoU_{1,1}}$	$\mathbf{mIoU_{1,0}}$	Forgetting	Avg. Learning Accuracy
Replay All	75	75.24	**67.52**	**6.79**	74.48
Naive Forgetting with CL	75	77.00	60.31	14.75	76.03
Mean Loss Replay with CL	75	77.8	64.04	11.02	76.42
Random Replay with CL	75	**78.28**	64.33	10.72	**76.66**

Table 9: Performance of the Cataract-1K trainings with CL and augmentation using the weights of the CaDIS training with CL and augmentation

The augmented version of the model outperforms the non-augmented one on both CaDIS and Cataract-1K datasets, as shown in Table 8. As shown in Table 9, simply adding

CL in the second training step does not, by itself, improve the forgetting score. Sample replaying remains essential to mitigate catastrophic forgetting. Additionally, mean loss replay did not outperform random replay in this experiment. We hypothesize that the inclusion of CL in CaDIS training alters the effectiveness of replay methods, indicating that mean loss may not be universally optimal as it is model-dependent. Furthermore, one of the three random replay trainings yielded a forgetting score as low as 8%, suggesting that there is a "better" subset of samples that can be replayed to reduce the forgetting score.

5.3.8 Contrastive Learning with Prototypes

Method	**mIoU$_{0,0}$**	**mIoU$_{0,1}$**
CL with Augmentation	**75.05**	**26.72**
CL-Prototypes with Augmentation	74.27	26.67

Table 10: Performance comparison of Mask2Former trained on CaDIS using standard CL with augmentation and CL-Prototypes with augmentation.

In order to test if the prototypes are able to improve CL trainings even further, both of the CaDIS and Cataract-1K CL trainings are done with prototypes. Prototypes are initially trained on CaDIS and the results can be seen in Table 10. The pre-trained anchors are transferred to the the Cataract-1K training where the mean loss replay strategy is used. The mean loss replay trainings in Table 11 indicate that, prototypes, with their usage as robust anchors, have more potential to mitigate the forgetting than the standard CL training. It is worth noting that adjusting the prototypes' learning rate in CaDIS and Cataract-1K trainings could potentially further decrease the forgetting score.

Method	**mIoU$_{0,0}$**	**mIoU$_{1,1}$**	**mIoU$_{1,0}$**	**Forgetting**	**Avg. Learning Accuracy**
Replay All (No CL)	75	75.24	**67.52**	**6.79**	74.48
Mean Loss Replay with CL	75	77.8	64.04	11.02	76.42
Mean Loss Replay with CL-Prototypes	74.27	**80.87**	63.71	10.56	**77.57**

Table 11: Performance of Mean Loss Replay with CL-Prototypes and augmentation, compared to Replay All (No CL) and Mean Loss Replay with standard CL and augmentation. In the Mean Loss CL-Prototypes training, weights of the CaDIS CL-Prototypes with augmentation are used.

6 Conclusions

In this study, we investigated various strategies to mitigate catastrophic forgetting in domain-incremental semantic segmentation. Our evaluation encompassed different replay strategies, knowledge distillation techniques, and the use of contrastive learning with and without augmentation. The findings are summarized as follows:

- **Contrastive Learning Effectiveness:** Contrastive learning proved to be highly effective in improving domain generalization and reducing catastrophic forgetting. Specifically, the integration of contrastive learning with replay strategies significantly enhanced performance metrics.
- **Trade-Off Between Forgetting and Performance:** Our experiments indicated a trade-off between minimizing forgetting and maintaining high performance on new tasks. When measures to reduce forgetting were implemented, there was often a slight decrease in performance on the new task's test set. This highlights the need to balance retention of old knowledge with the acquisition of new information.
- **Importance of Sample Replaying:** Sample replaying is a crucial component in overcoming catastrophic forgetting. Methods such as mean loss replay and random replay showed substantial improvements in retaining previously learned knowledge while learning new tasks. However, the effectiveness of replay strategies varied, indicating the need for careful selection based on specific model and task requirements.
- **Augmentation Strategies:** Among the augmentation methods tested, color jittering provided the best results in terms of domain generalization. Incorporating effective augmentation strategies during contrastive learning further improved model robustness and performance across different domains.
- **Knowledge Distillation Limitations:** The knowledge distillation techniques evaluated did not significantly mitigate catastrophic forgetting compared to replay methods. Even when combined with mean loss replay, these methods were insufficient to maintain high retention of previously learned knowledge, suggesting the need for alternative or improved distillation strategies.
- **Contrastive Learning with Prototypes:** Our prototypes trainings showed that, prototypes are powerful class representations and as pre-trained anchor vectors, they can improve the performance of the contrastve learning. However, prototype trainings require a hyperparameter search on their learning rate to slow down their shifts in the new domain.

In conclusion, our study demonstrates that contrastive learning, particularly when combined with effective sample replay strategies and augmentations, offers a robust solution to mitigate catastrophic forgetting in domain-incremental semantic segmentation. However, a careful balance between retaining old knowledge and acquiring new information is essential for optimal performance.

7 Future Research

Given the time constraints of this study, we were unable to explore all potential approaches for mitigating forgetting in continual learning. Our future research will focus on several promising directions:
Firstly, we plan to investigate the use of prototypes with contrastive learning, experimenting with both frozen and slowly updated prototypes for the classes.
Secondly, we observed that replay methods based on loss selection yield varying results depending on the model checkpoints used. To address this, we intend to explore data-dependent selection mechanisms, such as class-based sample selection, which may provide more consistent results.
Lastly, we aim to evaluate our methods and validate our findings on standard benchmarks for domain-incremental learning. This will help ensure the generalizability of our approaches. We also intend to pursue publication of our research results in peer-reviewed journals to contribute to the broader scientific community.

Bibliography

[1] Qi Bi, Shaodi You, and Theo Gevers. *Learning Content-enhanced Mask Transformer for Domain Generalized Urban-Scene Segmentation*. 2023. arXiv: 2307.00371 [cs.CV]. URL: https://arxiv.org/abs/2307.00371.

[2] Nicolas Carion et al. *End-to-End Object Detection with Transformers*. 2020. arXiv: 2005.12872 [cs.CV]. URL: https://arxiv.org/abs/2005.12872.

[3] Mathilde Caron et al. "Emerging Properties in Self-Supervised Vision Transformers". In: *CoRR* abs/2104.14294 (2021). arXiv: 2104.14294. URL: https://arxiv.org/abs/2104.14294.

[4] Fabio Cermelli et al. "Modeling the Background for Incremental Learning in Semantic Segmentation". In: *CoRR* abs/2002.00718 (2020). arXiv: 2002.00718. URL: https://arxiv.org/abs/2002.00718.

[5] Liang-Chieh Chen et al. *DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs*. 2017. arXiv: 1606.00915 [cs.CV]. URL: https://arxiv.org/abs/1606.00915.

[6] Ting Chen et al. *A Simple Framework for Contrastive Learning of Visual Representations*. 2020. arXiv: 2002.05709 [cs.LG]. URL: https://arxiv.org/abs/2002.05709.

[7] Ting Chen et al. "A Simple Framework for Contrastive Learning of Visual Representations". In: *CoRR* abs/2002.05709 (2020). arXiv: 2002.05709. URL: https://arxiv.org/abs/2002.05709.

[8] Bowen Cheng, Alexander G. Schwing, and Alexander Kirillov. *Per-Pixel Classification is Not All You Need for Semantic Segmentation*. 2021. arXiv: 2107.06278 [cs.CV]. URL: https://arxiv.org/abs/2107.06278.

[9] Bowen Cheng, Alexander G. Schwing, and Alexander Kirillov. "Per-Pixel Classification is Not All You Need for Semantic Segmentation". In: *CoRR* abs/2107.06278 (2021). arXiv: 2107.06278. URL: https://arxiv.org/abs/2107.06278.

[10] Bowen Cheng et al. "Masked-attention Mask Transformer for Universal Image Segmentation". In: *CoRR* abs/2112.01527 (2021). arXiv: 2112.01527. URL: https://arxiv.org/abs/2112.01527.

[11] Arthur Douillard et al. "PLOP: Learning without Forgetting for Continual Semantic Segmentation". In: *CoRR* abs/2011.11390 (2020). arXiv: 2011.11390. URL: https://arxiv.org/abs/2011.11390.

[12] Arthur Douillard et al. "Small-Task Incremental Learning". In: *CoRR* abs/2004.13513 (2020). arXiv: 2004.13513. URL: https://arxiv.org/abs/2004.13513.

[13] Evangello Flouty et al. "CaDIS: Cataract Dataset for Image Segmentation". In: *CoRR* abs/1906.11586 (2019). arXiv: 1906.11586. URL: http://arxiv.org/abs/1906.11586.

[14] Negin Ghamsarian et al. *Cataract-1K: Cataract Surgery Dataset for Scene Segmentation, Phase Recognition, and Irregularity Detection*. 2023. arXiv: 2312.06295 [cs.CV]. URL: https://arxiv.org/abs/2312.06295.

[15] Ian J. Goodfellow et al. *An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks*. 2015. arXiv: 1312.6211 [stat.ML]. URL: https://arxiv.org/abs/1312.6211.

[16] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV]. URL: https://arxiv.org/abs/1512.03385.

[17] Tobias Kalb and JÃ¼rgen Beyerer. *Principles of Forgetting in Domain-Incremental Semantic Segmentation in Adverse Weather Conditions*. 2023. arXiv: 2303.14115 [cs.CV]. URL: https://arxiv.org/abs/2303.14115.

[18] Tobias Kalb, Björn Mauthe, and Jürgen Beyerer. *Improving Replay-Based Continual Semantic Segmentation with Smart Data Selection*. 2022. arXiv: 2209.09839 [cs.CV]. URL: https://arxiv.org/abs/2209.09839.

[19] Tobias Kalb et al. "Continual Learning for Class- and Domain-Incremental Semantic Segmentation". In: *2021 IEEE Intelligent Vehicles Symposium (IV)*. 2021, pp. 1345–1351. DOI: 10.1109/IV48863.2021.9575493.

[20] Prannay Khosla et al. *Supervised Contrastive Learning*. 2021. arXiv: 2004.11362 [cs.LG]. URL: https://arxiv.org/abs/2004.11362.

[21] James Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks". In: *CoRR* abs/1612.00796 (2016). arXiv: 1612.00796. URL: http://arxiv.org/abs/1612.00796.

[22] Marvin Klingner et al. "Class-Incremental Learning for Semantic Segmentation Re-Using Neither Old Data Nor Old Labels". In: *CoRR* abs/2005.06050 (2020). arXiv: 2005.06050. URL: https://arxiv.org/abs/2005.06050.

[23] V7 Labs. *The Complete Guide to Panoptic Segmentation*. Accessed: 2024-07-19. 2023. URL: https://www.v7labs.com/blog/panoptic-segmentation-guide.

[24] Zhizhong Li and Derek Hoiem. "Learning without Forgetting". In: *CoRR* abs/1606.09282 (2016). arXiv: 1606.09282. URL: http://arxiv.org/abs/1606.09282.

[25] Tsung-Yi Lin et al. *Focal Loss for Dense Object Detection*. 2018. arXiv: 1708.02002 [cs.CV]. URL: https://arxiv.org/abs/1708.02002.

[26] Wei Liu et al. "SSD: Single Shot MultiBox Detector". In: *Lecture Notes in Computer Science*. Springer International Publishing, 2016, 21â€“37. ISBN: 9783319464480. DOI: 10.1007/978-3-319-46448-0_2. URL: http://dx.doi.org/10.1007/978-3-319-46448-0_2.

[27] Ze Liu et al. "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows". In: *CoRR* abs/2103.14030 (2021). arXiv: 2103.14030. URL: https://arxiv.org/abs/2103.14030.

[28] Jonathan Long, Evan Shelhamer, and Trevor Darrell. *Fully Convolutional Networks for Semantic Segmentation*. 2015. arXiv: 1411.4038 [cs.CV]. URL: https://arxiv.org/abs/1411.4038.

[29] Michael McCloskey and Neal J. Cohen. "Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem". In: *Psychology of Learning and Motivation* 24 (1989), pp. 109–165. URL: https://api.semanticscholar.org/CorpusID:61019113.

[30] Leland McInnes, John Healy, and James Melville. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. 2020. arXiv: 1802.03426 [stat.ML]. URL: https://arxiv.org/abs/1802.03426.

[31] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. "V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation". In: *CoRR* abs/1606.04797 (2016). arXiv: 1606.04797. URL: http://arxiv.org/abs/1606.04797.

[32] Youngmin Oh, Donghyeon Baek, and Bumsub Ham. *ALIFE: Adaptive Logit Regularizer and Feature Replay for Incremental Semantic Segmentation*. 2022. arXiv: 2210.06816 [cs.CV]. URL: https://arxiv.org/abs/2210.06816.

[33] Lorenzo Pellegrini et al. *Latent Replay for Real-Time Continual Learning*. 2020. arXiv: 1912.01100 [cs.LG]. URL: https://arxiv.org/abs/1912.01100.

[34] Theodoros Pissas et al. "Effective semantic segmentation in Cataract Surgery: What matters most?" In: *CoRR* abs/2108.06119 (2021). arXiv: 2108.06119. URL: https://arxiv.org/abs/2108.06119.

[35] Simon Reiß et al. "Decoupled Semantic Prototypes enable learning from diverse annotation types for semi-weakly segmentation in expert-driven domains". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 15495–15506.

[36] Shaoqing Ren et al. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. 2016. arXiv: 1506.01497 [cs.CV]. URL: https://arxiv.org/abs/1506.01497.

[37] 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.

[38] Olga Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: *CoRR* abs/1409.0575 (2014). arXiv: 1409.0575. URL: http://arxiv.org/abs/1409.0575.

[39] Hanul Shin et al. "Continual Learning with Deep Generative Replay". In: *CoRR* abs/1705.08690 (2017). arXiv: 1705.08690. URL: http://arxiv.org/abs/1705.08690.

[40] Onur Tasar, Yuliya Tarabalka, and Pierre Alliez. "Incremental Learning for Semantic Segmentation of Large-Scale Remote Sensing Data". In: *CoRR* abs/1810.12448 (2018). arXiv: 1810.12448. URL: http://arxiv.org/abs/1810.12448.

[41] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762.

[42] Gido van de Ven, Hava Siegelmann, and Andreas Tolias. "Brain-inspired replay for continual learning with artificial neural networks". In: *Nature Communications* 11 (Aug. 2020), p. 4069. DOI: 10.1038/s41467-020-17866-2.

[43] Wenguan Wang et al. *Exploring Cross-Image Pixel Contrast for Semantic Segmentation*. 2021. arXiv: 2101.11939 [cs.CV]. URL: https://arxiv.org/abs/2101.11939.

[44] Weijia Wu et al. *Continual Learning for Image Segmentation with Dynamic Query*. 2023. arXiv: 2311.17450 [cs.CV]. URL: https://arxiv.org/abs/2311.17450.

[45] Tete Xiao et al. *Unified Perceptual Parsing for Scene Understanding*. 2018. arXiv: 1807.10221 [cs.CV]. URL: https://arxiv.org/abs/1807.10221.

[46] Enze Xie et al. *SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers*. 2021. arXiv: 2105.15203 [cs.CV]. URL: https://arxiv.org/abs/2105.15203.

[47] Chun-Hsiao Yeh et al. *Decoupled Contrastive Learning*. 2022. arXiv: 2110.06848 [cs.LG]. URL: https://arxiv.org/abs/2110.06848.

[48] Bo Yuan and Danpei Zhao. *A Survey on Continual Semantic Segmentation: Theory, Challenge, Method and Application*. 2023. arXiv: 2310.14277 [cs.CV]. URL: https://arxiv.org/abs/2310.14277.

[49] Friedemann Zenke, Ben Poole, and Surya Ganguli. "Improved multitask learning through synaptic intelligence". In: *CoRR* abs/1703.04200 (2017). arXiv: 1703.04200. URL: http://arxiv.org/abs/1703.04200.

Appendix

A Segmentation Tasks

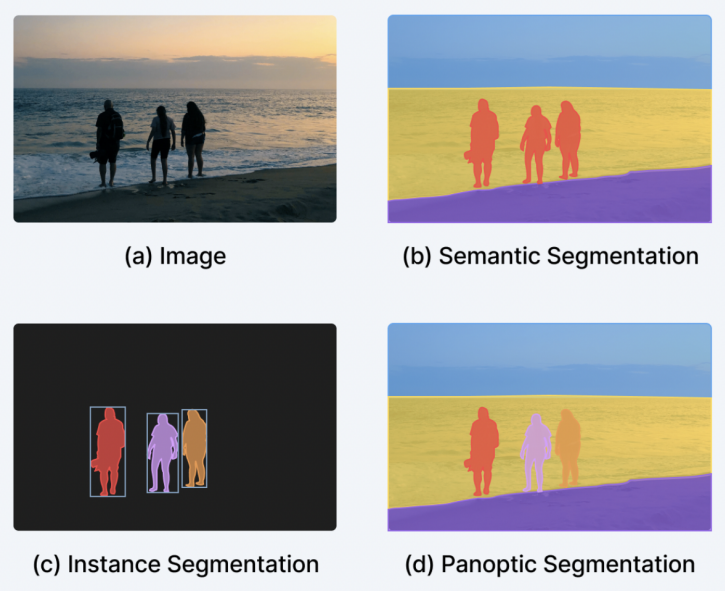


Figure 12: Difference between different types of image segmentations [23].

B Additional Notes on Contrastive Learning

The experiments where CL is used in both CaDIS and Cataract-1K, trainings are done under 30 epochs. The reasons are; the trainings took longer with the addition of CL, a budget constraint on GPU usage and the best performing models are usually obtained before the 30th epoch.
Secondly, we would like to elaborate on the pixel features taken from the pixel decoder. Together with the last hidden state and the intermediate features of the pixel decoder, the pixel decoder provides 4 image features with different sizes; [256,9,15], [256,17,30], [256, 34,60], [256,68,120]. These different sizes capture various levels of detail in the image. We upscale these matrices using bilinear interpolation to the the input image size; 270x480. Then, we compute the pixel level average of these 4 matrices to produce the final feature matrix. Each pixel is represented with a 256-dimensional vector hence, our pipeline doesn't require an additional projection head that is used in the original implementation [43]. We perform l_2-normalization on the pixel features before the contrastive loss.
Furthermore, it is worth mentioning that the total number of easy-positive and hard-negative pixels, in other words "n_views" for the given batch, is dynamically calculated for each batch. The formula can be found below:

$$\text{n\_views} = \min((\text{max\_samples} \;/\; \text{total\_classes}),\text{max\_views}) \tag{12}$$

where max_samples is 1024, max_views is 100 and total_classes changes according to the training type. "total_classes" also changes with the training scheme. If there are not

replayed samples in the batch, the "total_classes" is the sum of the unique classes (anchor classes) across all images in the batch. However, if it is a replay training, "total_classes" is the sum of the unique classes only from the new dataset images in the batch.
Finally, it's important to mention that the pixel sampling methods explained in the pixel-wise contrastive loss paper [43] are not implemented in their original project repository. Therefore, our adaptation of the pixel-wise contrastive loss and the hard-anchor sampling are based on their GitHub implementation: `https://github.com/tfzhou/ContrastiveSeg/tree/main`.