Few Shot Semantic Segmentation: Review and Implementation of a generalist Transformer based model.

Andrea Turchet[1]

[1]**Department of Artificial Intelligence & Cybersecurity, University of Klagenfurt,**
anturchet@edu.aau.at

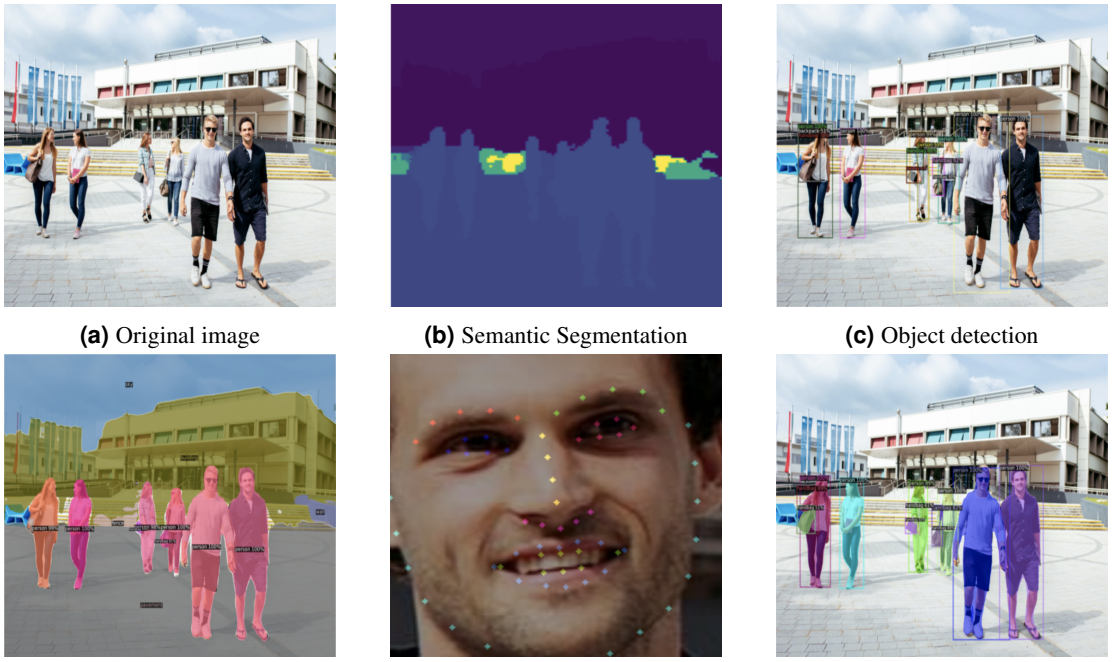
ABSTRACT

Through a comprehensive literature review and experimental analysis, this research investigates the adaptation and performance of state-of-the-art foundation models for semantic segmentation in specialized industrial domains using few-shot learning approaches. Specifically, will be examined the effectiveness of current SOTA models with few-shot learning methodologies for segmenting defects in semiconductor wafer images. The main purpose of this study is achieving an accurate semantic segmentation in industrial applications where labeled data is scarce and domain expertise is relevant. The literature review synthesizes current approaches and identifies gaps in existing methodologies, while the research explores various few-shot learning techniques integrated with foundational segmentation models, analyzing their efficacy in maintaining segmentation accuracy while reducing the requirement for extensive labeled training data.

Keywords: few shot learning, contrastive learning, generative models, variational auto encoders

INTRODUCTION

Computer Vision (CV) has become a fundamental discipline with far-reaching applications, from robotics and clinical analysis to video surveillance. A key task in this expansive field is semantic segmentation, which generates a segmentation mask by assigning a category label to every pixel in an image, identifying predefined subject categories. [23, 25, 37]. Contextualizing semantic segmentation requires understanding its relationship with other computer vision tasks, including image classification,instance segmentation, part segmentation, and object detection. Image classification focuses on comprehending the overall scene by assigning one or more labels to an entire image (see Fig. 1a). Object detection (see Fig. 1c) concentrates on predicting the location of objects, usually supplying bounding boxes for the identified objects. Parts segmentation (depicted in Fig. 1e) closely resembles semantic segmentation, with the aim of predicting pixel-level segmentation masks that cover distinct parts constituting the intended subject. Instance segmentation (see Fig. 1f) focuses on separating individual objects in an image, even those of the same kind, without the need to assign a specific category label. Finally, panoptic segmentation (Fig. 1d) seamlessly combines semantic segmentation with instance segmentation, predicting the category of each pixel and differentiating between individual instances of each class. Semantic segmentation occupies an intermediary position, as it provides sufficient image understanding to enable a large number of downstream applications, while still lacking deeper analysis and interpretation. The advent of Deep Neural Networks (DNN) and Convolutional Neural Networks (CNN) in recent years has revolutionized the Artificial Intelligence (AI) and Computer Vision (CV) research fields, allowing for the introduction of semantic segmentation models capable of impressive accuracy [45, 2, 60, 12, 92]. Nevertheless, semantic segmentation models exhibit a notable disadvantage: a reliance on extensive training datasets. Creating and labeling large datasets for semantic segmentation can require substantial resources. For example, labeling object instances for 80 classes in 328,000 images for the MS COCO [42] dataset required 55,000 worker hours. Collecting data this intensively is costly and time-consuming; this is especially true in agriculture and healthcare, where data is often variable and scarce, acquiring it requires specialized knowledge, and privacy concerns are paramount. To address this limitation, the Artificial Intelligence (AI) and Machine Learning (ML) communities have explored the concept of Few Shot Learning (FSL) [31, 32,



(a) Original image **(b)** Semantic Segmentation **(c)** Object detection

(d) Panoptic Segmentation **(e)** Parts Segmentation **(f)** Instance Segmentation

Figure 1. A comparison of different segmentation techniques applied to the same image: (a) Original image, (b) Semantic Segmentation (c) Object detection, (d) Panoptic Segmentation (e) Parts Segmentation, (f) Instance Segmentation. Original image source: University of Klagenfurt.

58, 80],which seeks to develop models that learn effectively from limited training data and generalize well to new, unseen data.

Applying the Few-Shot Learning (FSL) problem specifically to the semantic segmentation task has given rise to the field of Few-Shot Semantic Segmentation (FSS), which serves as the primary focus of this work. This comprehensive survey explores the main approaches used in few-shot semantic segmentation, where models learn to segment new classes from only 1 to 5 labeled examples. By exploring the key solutions proposed in the FSS literature, this work also describes the commonly used public benchmark datasets that have become important evaluation tools for assessing the performance of FSS models. Furthermore, this survey explain the technical implementation details of constructing models capable of performing Few-Shot Semantic Segmentation (FSS) in the context of a dataset containing images of semiconductor wafer defects and outlines effective strategies for designing models that can learn to accurately segment new defect classes from just a few labeled examples. This is of particular importance in the semiconductor manufacturing domain, where data variability and scarcity, along with the need for specialized domain knowledge, often hinder the acquisition of large-scale annotated datasets.

BACKGROUND

Few-Shot Semantic Segmentation (FSS) extends the concept of Few-Shot Learning (FSL) to the task of semantic segmentation; this allows models to learn to segment new objects or scenes with only a handful of labeled examples. Fig. 2 shows a timeline of all the approaches that will be discussed in the next sections of this work.

Few Shot Learning

Few-shot learning aims to generalize a model from only a small number of labeled examples. Let $\mathscr{X}$ be the input space and $\mathscr{Y}$ the output space, where we have a set of labeled examples $\mathscr{D}_{\text{train}} = \{(x_i, y_i)\}_{i=1}^{N}$ with $x_i \in \mathscr{X}$ and $y_i \in \mathscr{Y}$. In few-shot learning, the number of labeled samples N is typically very small, making generalization challenging [80].

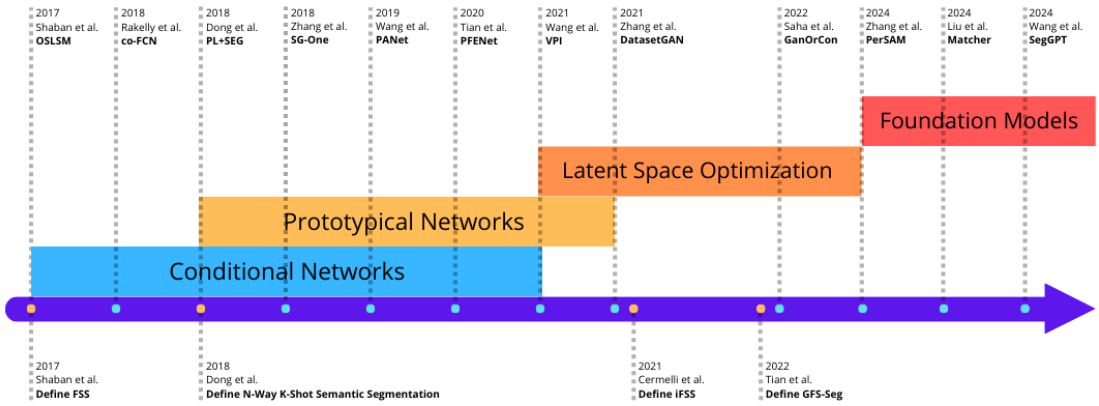


Figure 2. Timeline showing the development of the FSS research area.

Problem Setup

Formally, in few-shot learning, we often have:

- **Support Set**: A small labeled dataset $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{K}$, where K is small, usually in the range of 1 to 10.
- **Query Set**: An unlabeled dataset $\mathcal{Q} = \{(x_j)\}_{j=1}^{M}$ for which the model needs to predict the labels based on the support set.

Semantic Segmentation

Semantic segmentation is the task of assigning a predefined category label to each pixel in an image, as shown in Fig. 1b compared to the original image in Fig. 1a. The goal is to partition the image into mutually exclusive subsets, where each subset represents a meaningful region of the original image. Few-shot semantic segmentation aims to classify each pixel in an image into semantic categories using only a small set of labeled examples. Unlike traditional segmentation, where a large annotated dataset is available, few-shot segmentation must generalize from limited labeled samples. Let $\mathcal{X}$ represent the space of images and $\mathcal{Y}$ the space of segmentation masks, where each pixel is assigned a label from a set of classes $\mathcal{C} = \{c_1, c_2, \ldots, c_N\}$.

Problem Setup

In few-shot segmentation, we have:

- **Support Set**: A small dataset $\mathcal{S} = \{(I_i, M_i)\}_{i=1}^{K}$, where each $I_i \in \mathcal{X}$ is an input image, and $M_i \in \mathcal{Y}$ is the corresponding segmentation mask. The mask M_i assigns each pixel in I_i a label from the class set $\mathcal{C}$.
- **Query Set**: An unlabeled set of images $\mathcal{Q} = \{I_j\}_{j=1}^{M}$, for which we aim to predict the segmentation mask $\hat{M}_j$ based on information from the support set.

In N-way K-shot segmentation, we focus on N distinct classes, with K annotated examples per class in the support set. The goal is to learn a segmentation function $f : \mathcal{X} \to \mathcal{Y}$ that produces accurate pixel-wise predictions for each query image $I_j \in \mathcal{Q}$, using only the limited labeled samples in $\mathcal{S}$ [63]. With the limited availability of labeled samples, conventional algorithms often can be inadequate in training FSS models effectively. To make the most of these few examples, a majority of studies [6, 16, 38, 46, 50, 55, 63, 65, 70, 77, 83, 86] utilize episodic training, a meta-learning or "learning-to-learn" method. Another frequently used approach is transfer learning, which uses the knowledge gained by pre-training parts of the model on different datasets or tasks. As these techniques are fundamental, the following section provides an overview of episodic training in FSS and details how pre-training can be adapted for FSS.

Episodic Training

Unlike conventional training paradigms, few-shot segmentation relies on episodic training to improve generalization under limited data conditions. In episodic training, the model is trained on tasks (or "episodes") that simulate the few-shot evaluation process, helping the model learn how to learn from limited labeled samples [72].

Episodic Training Setup

In episodic training, the model learns through a sequence of "episodes" rather than training directly on the entire dataset. Each episode is structured as a few-shot task and consists of:

- **Episodes**: Each episode simulates a few-shot learning scenario, consisting of a small support set and a query set tailored for specific, randomly selected classes.

Key differences between episodes and batches: Traditional batches randomly sample images from the entire dataset without structuring them into specific few-shot tasks, making them less task-specific.
Each episode is structured explicitly as a few-shot task and typically involves the following steps:

1. Select a subset of N classes randomly from the full class set $\mathscr{C}$.
2. Construct a **support set** $\mathscr{S} = \{(I_i, M_i)\}_{i=1}^{K}$, providing K labeled examples per selected class.
3. Create a **query set** $\mathscr{Q} = \{I_j\}_{j=1}^{M}$, containing unlabeled or partially labeled images that the model must segment using information inferred from the support set.
4. Train the model to predict segmentation masks for the query set, conditioned only on the few labeled examples in the support set.
5. Update model parameters based on the segmentation performance on the query set.
6. Repeat the process with new episodes, continuously varying the classes and examples to encourage robust generalization.

While traditional training uses batches to stabilize gradient estimates and improve computational efficiency, episodic training can also benefit from batching. In this context, a mini-batch refers to a collection of multiple episodes processed together in a single training iteration. This approach combines the advantages of episodic training with the computational benefits of batching [11].

Why Episodic Training is Used

The effectiveness of episodic training for few-shot learning is due to its alignment with the evaluation conditions:

- **Mimics Test Conditions**: By training on episodes that simulate few-shot tasks, the model learns to generalize from limited examples, preparing it for the actual few-shot segmentation challenges it will face during evaluation.
- **Task-Specific Adaptation**: Episodic training encourages the model to adapt to different tasks by conditioning on the support set, helping the model develop a meta-learning capability. This adaptation promotes effective segmentation even for classes unseen during training.
- **Improves Representation Learning**: Episodic training forces the model to learn transferable representations, as it must segment classes in the query set based solely on information from the support set. This results in a feature space that captures high-level semantics and generalizes across diverse tasks.

Benefits of Episodic Training for Few-Shot Segmentation

Episodic training confers several advantages, it is possible to found in the literature that by learning from diverse tasks with different classes in each episode, the model becomes robust to class changes, making it effective for segmenting unseen classes during testing. Episodic training builds a transferable feature space by requiring the model to segment images based on limited labeled data, thereby learning high-level representations. This training approach also prepares the model to adapt quickly to specific tasks, which is essential in few-shot scenarios where labeled data is sparse.

Transfer Learning

In few-shot semantic segmentation, the goal is to segment new categories with only a few labeled examples. Transfer learning is a technique that hold knowledge from a large, labeled source dataset to improve model performance on a target dataset with limited labeled examples. By transferring learned representations or parameters from a source domain, the model can generalize better in the low-data regime of the target domain. Although the reliance on a large dataset for pretraining may appear at odds with the low-data requirements of few-shot settings (FSS), it's important to note that "few shots" refers only to the new class the model is tasked with learning. Thus, as long as the semantic classes used in pretraining are distinct from those to be learned with limited examples, pretrained feature extractors serve as a valid form of transfer learning in this context [19, 50, 90]. Transfer learning in few-shot segmentation involves two domains:

- **Source Domain** $\mathscr{D}_S$: A dataset with images and dense pixel-wise labels for a large number of classes $\mathscr{C}_S = \{c_1, c_2, \ldots, c_{N_S}\}$, where N_S is large.
- **Target Domain** $\mathscr{D}_T$: A dataset with limited labeled examples for a set of target classes $\mathscr{C}_T = \{c_{N_S+1}, \ldots, c_{N_S+N_T}\}$, where N_T is small.

In few-shot segmentation, the target domain has a support set $\mathscr{S}_T = \{(I_i, M_i)\}_{i=1}^{K}$, where each image I_i has a corresponding segmentation mask M_i. We aim to predict segmentation masks for a query set $\mathscr{Q}_T = \{I_j\}_{j=1}^{M}$ in the target domain, based on knowledge transferred from the source domain.

Why Transfer Learning is Used in Few-Shot Segmentation

Transfer learning is effective in few-shot segmentation due to the following reasons:

- **Representation Learning**: Pre-training on the source domain $\mathscr{D}_S$ allows the model to learn a feature representation that captures essential patterns (e.g., edges, textures, object parts) relevant across domains. These representations can then be fine-tuned in the target domain with minimal labeled data.
- **Parameter Initialization**: Transfer learning provides a well-initialized set of parameters for the segmentation model. Rather than learning from scratch, the model starts from a set of weights that encode general semantic knowledge, making fine-tuning more effective and less data-dependent.
- **Domain Adaptation**: Although the classes in $\mathscr{D}_S$ and $\mathscr{D}_T$ differ, the model can use shared visual features (e.g., common shapes or textures) learned in the source domain to aid in segmenting novel classes in the target domain.

Benefits of Transfer Learning in Few-Shot Segmentation

There are several different benefits offered; first of all the efficient utilization of labeled data derived from the use of a large labeled source dataset, transfer learning reduces the dependence on extensive labeled data in the target domain. The rapid adaptation to new classes provides a strong initialization, enabling the model to adapt to new classes in the target domain quickly and with few examples. Transfer learning also helps the model to capture general features that are useful across domains, making it more robust to new and unseen classes.

In-Context Learning

The emergence of in-context learning within the field of computer vision has been a notable novel advancement. Bar et al. [3] first introduced this approach by reformulating in-context learning as an image inpainting task. In their framework, models are presented with a concatenated image comprising multiple exemplar images and a novel input image containing a designated region to be inpainted. The model's objective is to infer the missing content within this region based on the contextual information provided by the exemplar images. The authors empirically validated the efficacy of this in-context learning paradigm across several visual tasks, including foreground segmentation, single object detection, and image colorization, demonstrating its potential for zero-shot or few-shot generalization. Formally, given a target image $\mathbf{I} \in \mathbb{R}^{3\times H\times W}$ where H and W represent the height and width, the goal is to segment a binary mask $\mathbf{y} \in \mathbb{R}^{\{0,1\}^{H\times W}}$ conditioned on K in-context examples $\{\mathbf{x^r}\}^K = \{(\mathbf{I^r}, \mathbf{y^r})\}^K$, where $\mathbf{I^r}, \mathbf{y^r}$ are the image and ground-truth mask of an in-context example. Building upon this foundational work, Painter [78] extended in-context learning to a broader range of computer vision problems by leveraging masked image modeling (MIM) on continuous pixel patches. In their method, models are trained on supervised datasets using an MIM objective, where random portions of the input image are masked, and the model is tasked with reconstructing the masked regions. Painter demonstrated the competitiveness of this approach on seven distinct vision tasks, suggesting that in-context training with MIM can enhance model adaptability and transfer learning capabilities. Further specializing the concept, SegGPT [79] concentrated on the application of in-context learning to image segmentation. A core contribution of SegGPT is the introduction of a random coloring scheme applied to the input images; This effectively forces the model to rely on contextual cues from the provided examples to accurately perform the segmentation task, as the color information alone becomes insufficient for delineating object boundaries. This method highlights the importance of context in driving accurate segmentation performance within the in-context learning framework.

METHODOLOGY REVIEW

The field of Few-Shot Segmentation (FSS) is inherently multifaceted, allowing for exploration from various perspectives. This section aims to highlights some of the most influential and significant studies in this domain. To enable a flexible approach to various research directions, this work propose to explain three core strategies: conditional networks, prototypical learning, and latent space optimization. Latent space optimization can be further divided into two specialized approaches: the generation of synthetic datasets, such as through Generative Adversarial Network (GAN) encodings, and methods based on contrastive learning. Additionally, this report will discuss the novel opportunities presented by recent foundation models and their potential applications in addressing FSS tasks.

Conditional networks

Conditional networks represent a foundational methodology in the field of Few-Shot Segmentation (FSS), first introduced by Shaban et al. [63] and Rakelly et al. [55] these models employ a two-branch architecture: the conditioning branch and the segmentation branch. Fig. 3 refers to this specifc tipology. The conditioning branch processes a support set S, producing a parameter set θ. Simplifying, this θ acts as an intermediary that informs the segmentation branch, which subsequently takes a query image I_q and predicts a segmentation mask $\hat{M}_q$ using both θ and a feature representation of I_q. The segmentation branch involves a feature extraction process, wherein a dense feature volume F_q is generated from the query image. Each spatial location in F_q undergoes computation, influenced by θ, to output the mask $\hat{M}_q$. More formally:

$$\theta = g(S), \quad F_q = \phi(I_q), \quad \hat{M}_q = h(F_q, \theta), \tag{1}$$

where g represents the conditioning branch, ϕ is the feature extractor, and h denotes the segmentation branch.

The earliest conditional networks incorporated pre-trained feature extractors and straightforward fusion techniques for parameterization. In the research conducted by Rakelly et al. [55], the authors experimented with two distinct forms of θ: one derived as a feature volume from the support set and the other as a set of linear classifier weights. Their findings revealed that directly using fused feature volumes led to more accurate outcomes compared to simpler linear classifiers. Shaban et al. [63] further refined this process by applying pixel-wise logistic regression using θ, which improved segmentation precision. Later, Lu et al. [46] introduced transformers to refine θ predictions. In this approach, θ served as query vectors in the transformer module, while the feature vectors of the query image acted as keys and values. Such transformer-based advancements allowed for dynamic contextualization of feature representations, enhancing segmentation performance; recognizing the value of diverse parameter sets, Zhang et al. [89] proposed the use of three distinct embeddings— Peak, Global, and Adaptive. Each of these embeddings captures unique facets of the target class, Peak Embedding focuses on the most discriminative features, identifying the key areas of interest within the image and understanding its broader context. Last, Adaptive Embedding employs self-attention mechanisms to dynamically adjust the focus based on input content. By combining these different ways of representing information, the model becomes more versatile and can handle complex tasks that would be difficult for a single approach. The effectiveness of conditional networks heavily relies on the interplay between θ and the segmentation branch. The parameter θ must encapsulate sufficient information from the support set to guide the segmentation process effectively. However, learning this complex relationship is difficult, especially when the information from the support set is too detailed or unrelated to the image we want to segment.

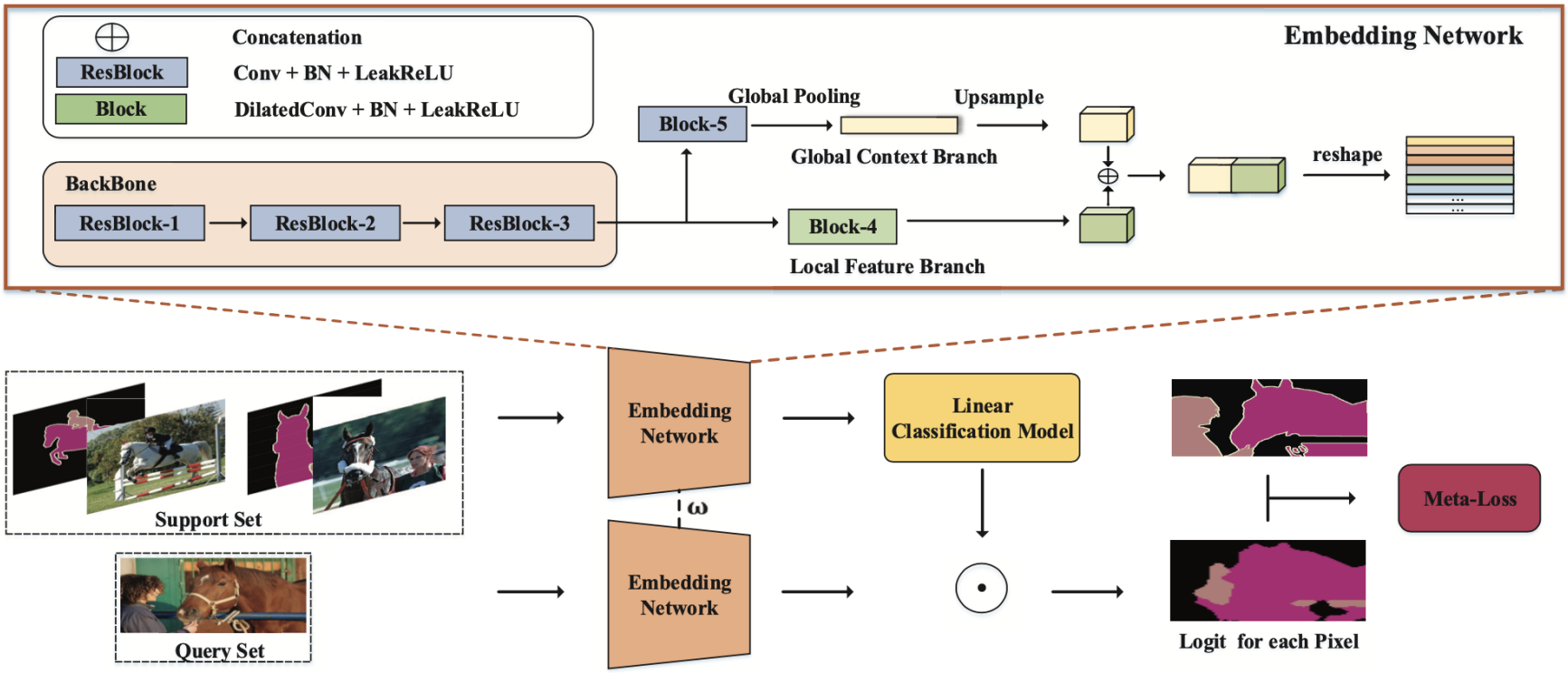


Figure 3. Pipeline of MetaSegNet for 2-way semantic segmentation. This is based on a Conditional network[69].

Prototypical Networks

Prototypical Networks, originally developed for Few-Shot Classification (FSC) [74], have been adapted and widely adopted for Few-Shot Segmentation (FSS). Basically, this approach uses metric learning to address the challenges of limited data in FSS scenarios. The core concept of Prototypical Networks is based on the idea that similar objects cluster together in an embedding space, while dissimilar objects are farther apart (Fig. 4 as reference). In this framework, a class prototype is created by computing the centroid of all embedded support set images belonging to that class and during inference, new images are classified by finding the nearest prototype in the embedding space. Dong and Xing [16] extended this concept to FSS by treating segmentation as a pixel-wise classification task. In their approach, each pixel of a query image is projected into a learned feature space and then assigned the label of the closest prototype. This adaptation has proven effective and has been widely adopted in subsequent FSS research [77, 86, 44, 40].

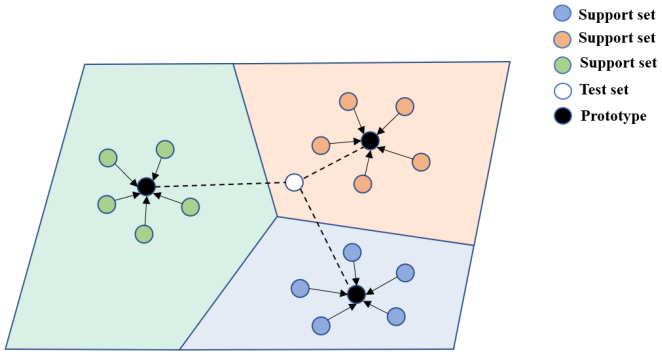


Figure 4. ProtoNet schematic diagram. Adapted from Snell et al. [67].

Masked Average Pooling (MAP) emerged as a response to the limitations of traditional pooling methods in preserving spatial information necessary for tasks like segmentation [90]. Unlike global average pooling or max pooling, MAP introduces a masking mechanism that allows for selective feature aggregation. One of the key advantages of MAP is its ability to preserve spatial information. By selectively focusing on relevant regions of the feature map, MAP retains important spatial cues that might be lost in global pooling operations; this is particularly beneficial in tasks like segmentation, where spatial relationships are crucial. While ResNet50, ResNet101 [28], and VGG16 [66] are commonly used as backbone networks for feature extraction, there is no clear consensus on the optimal choice. Each backbone offers different trade-offs in terms of depth, computational complexity, and feature representation capacity. The masking mechanism in MAP serves as an effective filter for irrelevant information or noise in the feature maps and by emphasizing relevant features and suppressing others, MAP contributes to cleaner, more robust feature representations. Another design choice involves in the selection of distance metric in measuring similarity between prototypes and query features in few-shot segmentation tasks. While Snell, Swersky, and Zemel [67] initially proposed using Bregman distance divergences, such as squared Euclidean distance, recent empirical evidence suggests that cosine distance may be more effective [77]. Concisely, cosine distance offers several advantages, for example it provides more stable results during the optimization process and unlike Euclidean distance, cosine similarity is bounded between -1 and 1, making it more suitable for optimization. Due to these benefits, cosine distance has become the preferred metric in many subsequent works [44, 70, 77, 86]. The adoption of cosine distance represents an important evolution in the design of few-shot segmentation models, potentially leading to more robust and accurate segmentation results. Prototype-based models in FSS have limitations when dealing with subjects composed of multiple sub-parts. For example, a dog's head, body, and limbs have distinct features that may not be adequately captured by a single prototype. Recognizing this limitation, Yang et al. [85] introduced a prototype mixture model, employing multiple prototypes to more comprehensively represent a class, thereby capturing distinct macro-features. Extending this idea, Zhang et al. [87] and Wang et al. [73] adopted a graph-based approach. They constructed a fully connected graph where nodes represent feature vectors from both support and query images, and edge weights denote the similarity between two features, then these connection weights are utilized for pixel-level classification.

Another work from Li et al. [40] addresses to the issue of background class confusion. They proposed a method with a number of prototypes greater than the number of classes with the introduction of a pseudo-class for areas of the image that correspond to high activation but are not labelled. This approach allows the model to exploit better the limited support set by inferring a new class. For the issue regarding the prototype separation, Okazawa [51] penalized high similarity between prototypes; this is useful to reduce the risk of overly similar prototypes in the sparse few-shot latent space. Fan et al. [20] updated prototypes using high-confidence query features, potentially incorporating less-represented information. the use of that features, while representing high-confidence parts of the subject in the query, enable the incorporation of underrepresented features within the query. Recognizing that different classes may share similar middle-level features, Wu et al. [83] introduced a Meta-class Memory Module. Basically, this module aligns query and support features at the middle level to enhance final segmentation mask prediction. By learning meta-class embeddings, the module effectively utilizes shared features across different classes. Most of the works discussed above share the similar issue; the choice of an optimal backbones for feature extraction.

Latent space optimization

The efficacy of few-shot learning, particularly in the context of semantic segmentation, depend on the model's ability to extract and leverage meaningful representations from limited data. Consequently, the research has focused on optimizing the latent space within deep learning models to enhance their generalization capabilities in low-data regimes. This section reviews methodologies that explore latent space optimization as a core strategy for improving few-shot semantic segmentation performance. These methods typically aim to learn a latent space that is both discriminative, enabling clear separation of semantic classes, and adaptable, allowing the model to quickly adjust to new classes or visual domains with minimal examples.

Generative Models

In object segmentation applications, particularly those involving fine-grained distinctions such as part segmentation, availability of large amount of data often contrasts with a scarcity of precise label information. While large-scale datasets of objects like faces or cars (e.g., from public sources) are readily available, generating detailed segmentation masks for their constituent parts remains a labor-intensive process. To mitigate this labeling bottleneck, recent research has explored the application of generative models within the FSS framework [61, 71, 86]. These methods are based on the observation that high-fidelity generative adversarial networks (GANs), such as StyleGAN [33] and StyleGAN2 [34], implicitly encode rich semantic and geometric information within their internal representations, effectively disentangling latent factors of variation. Empirical evidence provided by Karras [33], demonstrate that interpolations between latent codes of different images yield synthetic images that maintain qualitative fidelity. This suggests that the internal representation of a GAN-generated image potentially encapsulates more informative features than those extracted through conventional methods, thereby offering a promising avenue for improving segmentation performance. Using this insight, Zhang et al. [91] and Tritrong, Rewatbowornwong, and Suwajanakorn [71] harness the inherent regularity of StyleGAN and StyleGAN2 latent spaces for part segmentation prediction; their proposed architectures incorporate an FSS module that predicts segmentation masks based on the GAN's internal representation of a generated image. The operational pipeline of these models consist in a two-stage process. First, given a large supply of unlabeled images from the target domain, a GAN is trained to optimize its latent space, effectively learning the underlying data distribution. Subsequently, a training set for the FSS module is curated by sampling k images generated by the GAN, along with their corresponding internal representations. These k images are then manually annotated to provide ground-truth segmentation masks, forming the basis for supervised training of the FSS module. Upon completion of the model training, the authors of [71] further introduce a latent code optimization step. This involves iteratively refining the GAN's latent code to generate an image that closely resembles a given test image, while simultaneously recording the corresponding internal representation. Finally, both the generated image and its internal representation are fed into the trained segmentation module to predict the part segmentation mask for the test image, scheme in Fig. 5 . This method proficiently utilizes the learned representation of the GAN to enhance segmentation accuracy in a few-shot learning framework.

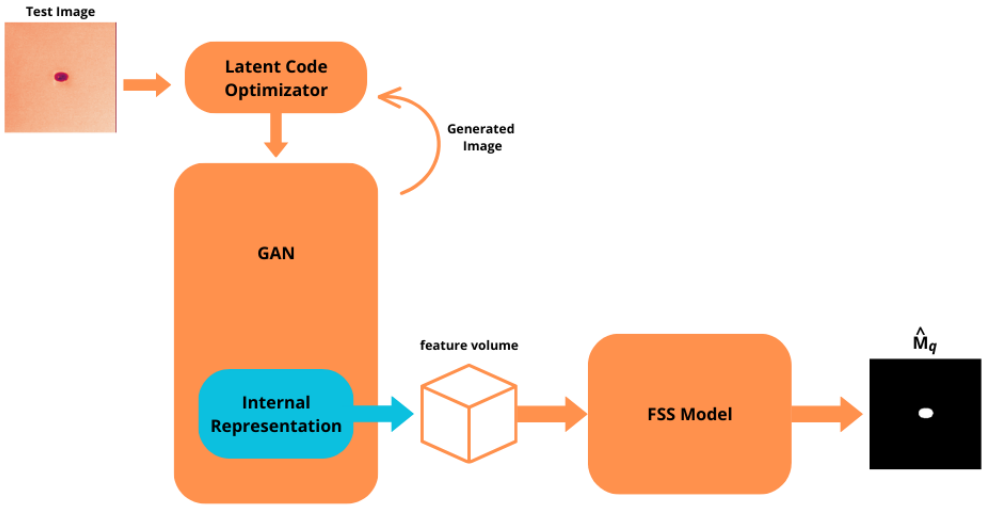


Figure 5. Rapresentation of a GAN performing few-shot segmentation by first optimizing its latent code to generate an image that closely resembles the test input. The internal representation of the GAN, effectively a feature volume encoding the generated image, is then extracted. This feature volume serves as input to an FSS model, which is responsible for generating the final predicted segmentation mask, $\hat{M}_q$

It is necessary to note that the authors explain that the latent code optimization is computationally expensive and also, the generated images can still be dissimilar compared to the test image, falling in a bad performance model. To address this limitations, the papers [71, 91] propose a strategy that uses pre-trained generative adversarial networks (GANs) for synthetic data generation, as illustrated in Fig. 6. This process involves sampling novel images from the trained GAN and subsequently employing the FSS module to predict corresponding part segmentation masks. The resulting synthetic dataset, with inside the GAN-generated images paired with inferred masks, can then be utilized to train any standard, off-the-shelf segmentation model. Experiments, presented in [71], demonstrate the efficacy of this approach, showing that generating a new part segmentation dataset in this way circumvents the need for the computationally expensive latent code optimization step during inference, while simultaneously enhancing segmentation performance.

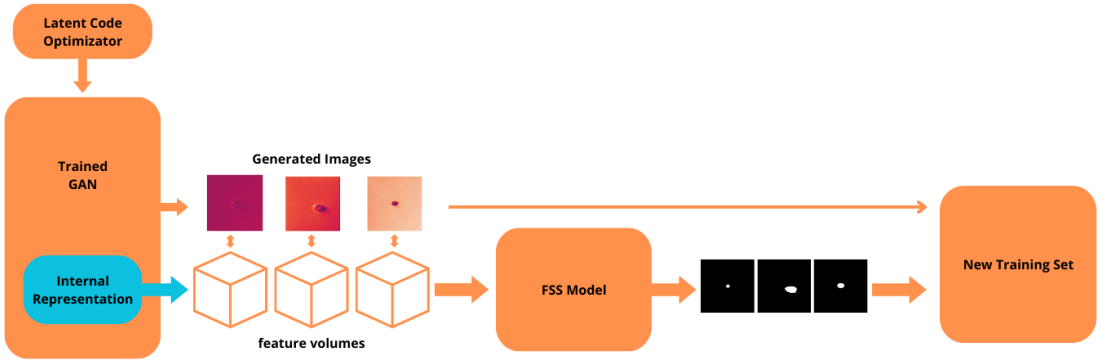


Figure 6. The synthetic dataset is created using a generative model with a limited annotation effort. A GAN is trained on the target dataset. Few synthetic images are labeled and, along with their internal representations, train a FSS module. During inference, unlimited images and internal representations generated by the GAN are used by the FSS module to create masks. This new synthetic dataset, comprising generated images and masks, can train any existing segmentation model.

While GANs offer a powerful tool for feature extraction [61, 71] and synthetic data generation to alleviate data scarcity, their training procedures can be complex and resource-intensive. The diversity of approaches within this family of methods makes it challenging to define a standardized training

protocol, unlike conditional and prototypical networks where episodic learning, as described in Section Background, is commonly adopted. Furthermore, GAN-based FSS models exhibit flexibility in the number of shots utilized for training, deviating from the conventional 1 or 5-shot settings typical of conditional and prototypical networks. For instance, Zhang et al. [91] employ 20 GAN-generated images to train a multi-layer perceptron (MLP) based FSS module. Similarly, Tritrong, Rewatbowornwong, and Suwajanakorn [71] experiment with 1, 5, and 10 shots for FSS module training. Although GANs have demonstrated promise in FSS, particularly when an abundance of unlabeled data is available, the complexities and computational demands associated with their training have motivated the exploration of alternative approaches. Specifically, the benefits derived from the rich internal representations and inherent regularities of GANs are often counterbalanced by challenges, such as the computationally intensive latent code optimization step. To address this, the next section will explore research that investigates simpler architectures for FSS, with the aim of achieving comparable performance without incurring the substantial overhead associated with training and deploying GANs.

Contrastive Learning

To mitigate the challenges associated with GAN-based approaches described in the section before, Saha, Cheng, and Maji [61] proposed the integration of contrastive learning for training the feature extractor. Their work demonstrates that contrastive learning enables the backbone network to consistently produce latent representations that surpass those of GAN-based models in performance, without introducing additional model complexity. Contrastive learning, as detailed in [1, 13, 17], involves training a model on a self-supervised task in order to develop invariance to augmentations in its representations. This is typically accomplished by constructing a dataset of image pairs where each pair comprises one image from the original dataset and a second image that is either a transformed version of the first (positive pair) or a distinct image (negative pair). The model is then trained to discern whether the two images within a pair depict the same underlying subject, effectively learning to map augmented versions of the same image to similar latent representations. Following this self-supervised pre-training, the model can be employed as a feature extractor. The test image is then fed into the network, and its resultant internal representation is recorded. Saha, Cheng, and Maji [61] utilize the feature volume extracted via this contrastive learning paradigm to predict part segmentation masks, employing a U-Net [60] inspired architecture. Notably, this model achieves marginally superior performance compared to the GAN-based model proposed by [91], while at the same time exhibiting a significantly reduced training complexity.

Variational Autoencoders

In the field of few-shot segmentation (FSS), prototypical networks have served as a foundational approach for representation. Is it worth to note that the deterministic nature of their feature extractors limits their effectiveness, especially when confronted with significant intra-class variations and noise, as is often the case in one-shot scenarios. The process of compressing an entire class into a single deterministic prototype vector can also discard valuable structural information about the objects present in the scene. Recognizing the need for a more regularized latent representation, Wang et al. introduced an innovative approach based on variational autoencoders (VAEs) [35, 59]. Formally, in VAEs, regularizations forces distributions to be close to a standard normal distribution; however, the log-likelihood can force the autoencoder to prefer other latent regions to minimize the loss function. For this reason a VAE can be non-linear mixture of Gussian distributions. Unlike traditional prototypical networks, their model embraces a probabilistic framework, learning a probability distribution over potential prototypes for each given class. This allows for a more comprehensive and adaptable encoding of object variations. Sun et al. [68] later augmented this model with a variational attention mechanism. This addition serves to highlight the foreground elements of the test image, leading to further improvements in overall segmentation performance.

Foundational models

The advent of foundation models has marked a turning point in the evolution of artificial intelligence. These powerful, adaptable models, trained on massive datasets, are capable of tackling a wide range of downstream tasks. While the Natural Language Processing (NLP) domain has been at the forefront, producing models like BERT [15] and GPT [7, 53] , the field of Computer Vision (CV) is rapidly growing, featuring foundation models like CLIP [54] and the innovative Segment Anything Model (SAM) [36]. By employing contrastive learning, CLIP aligns the embeddings of images and their corresponding textual descriptions, encouraging a cross-modal understanding that fuels a multitude of tasks, from

classification and retrieval to the more creative realm of image generation. Meanwhile, Kirillov et al. [36] have introduced SAM, a foundation model specifically designed for image segmentation, built upon an innovative promptable segmentation framework. Trained on an unprecedented dataset of 11 million image-mask pairs, meticulously collected through a novel data engine, SAM possesses the remarkable ability to segment virtually any object within a visual scene. Its versatility is further enhanced by its ability to accept a wide variety of prompts, both sparse (such as points, boxes, or text) and dense (like masks), returning the corresponding segmentation mask. This flexibility positions SAM as a truly generalist vision foundation model for image segmentation. However, it's important to note that SAM, in its current form, lacks the capacity to identify the specific class of each segmented object. These models provide an opportunity for FSS, enabling segmentation of classes without custom datasets. Three primary strategies have emerged: prompt engineering, multimodal approaches, and the development of generalist models.

Prompt engineering

Foundation models like SAM, while remarkably capable, often require additional guidance in the form of prompts to delineate specific object classes. The fact that SAM is meticulously designed to understand prompts in diverse formats, including point boxes, text, and masks,has sparked extensive research into prompt engineering. An interesting example of this approach is Matcher [43], introduced by Liu et al., as a model for one-shot segmentation that uses SAM as its foundational model. Matcher's modus operandi involves embedding both the support and query images using the DINOv2 feature extractor [52]. Recognizing the inherent correspondence between spatial locations in the feature volume and patches within the images, Matcher meticulously computes a similarity matrix between features, pinpointing similar patches across the images. High similarity score patches are chosen as SAM prompts, whose predictions are aggregated to produce the final segmentation output. Similarly, PerSAM [88], a training-free personalization approach tailored for SAM, ingeniously customizes it using one-shot data, which consists of a reference image and a rough mask delineating the desired concept. PerSAM initiates its process by generating a location confidence map for the target object in the test image, predicated on feature similarities. It then selects two points as positive-negative location priors, guided by confidence scores. These points are encoded as prompt tokens for SAM's decoder, orchestrating the segmentation process. Both Matcher and PerSAM are heavily reliant on the feature volumes of the reference and test images to generate effective prompts, underscoring the critical importance of flexible and informative embeddings. The focus on embeddings make this approach similar to the idea of prototypical learning, as discussed in Subsection Prototypical Network. Building upon these principles, SAM 2 [57] significantly advances prompt-guided segmentation, particularly in the video domain. For images, SAM 2 essentially functions similarly to SAM, but with key improvements that boost its performance and efficiency. SAM 2's image encoder is smaller yet more effective, leading to faster processing—about 6x faster than SAM—while also being more accurate in image segmentation benchmarks, the architecture's scheme is explained in Fig. 7. This efficiency comes from a hierarchical image encoder, pre-trained with the MAE method, which enables the use of multiscale features during decoding. In image segmentation, when using SAM 2, the memory is left empty and the model behaves like SAM. The advancements in SAM 2, coupled with SAM's versatile prompt understanding, have driven significant research in prompt engineering. This is exemplified by models like Matcher, which leverages SAM as its foundation. Although SAM 2 was designed with video segmentation in mind, its architectural improvements significantly benefit image segmentation tasks, demonstrating that prompt engineering techniques developed for SAM remain highly relevant and adaptable to its successor.

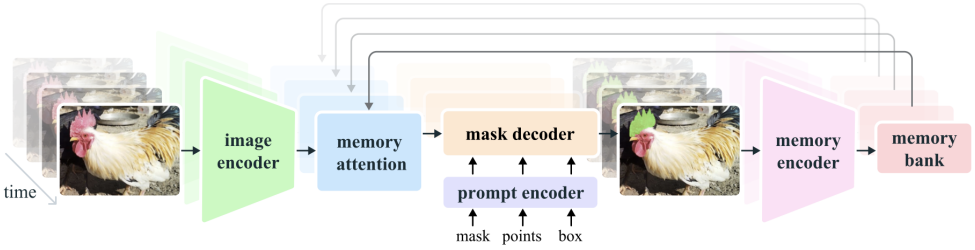


Figure 7. Architecture of SAM2 model, taken from [57]

Multimodal Approaches

The advent of CLIP [54] has significantly improved the research in the multimodal approach of FSS domain. Trained on a vast and diverse dataset, CLIP exhibits remarkable flexibility, enabling zero-shot classification of images. This inherent adaptability has inspired numerous researchers to develop zero-shot segmentation models that accept textual descriptions of the target subject as prompts. Many of these endeavors [47, 81, 84, 93] have adopted an encoder-decoder architecture. Typically, the encoder utilizes a pre-trained CLIP model to extract aligned visual and text embeddings; the alignment of these embeddings, despite originating from inputs of disparate natures, permits the decoder module to predict segmentation masks by focusing its design on the inner product between the text embeddings and the visual embeddings. While multimodal models that leverage text prompts alleviate the need for expansive segmentation datasets, it is important to acknowledge that for certain application-specific domains, describing targets textually may present challenges. For instance, textual ambiguity could, in some cases, render this approach less than ideal. In the Subsection about techniques used in the In-context learning, which were primarily developed within the NLP domain, the generalist models can rapidly assimilate new tasks with a minimal number of examples or prompts. This adaptability extends its reach into Few-Shot Learning (FSL) field, enabling their application in FSS tasks. However, while In-Context Learning has increased in the NLP domain, its exploration in the field of computer vision remains nascent, primarily due to significant disparities between the two domains. Unlike NLP tasks, which revolve around discrete language tokens, CV tasks encompass a diverse array of output representations, posing significant challenges in defining effective task prompts.

Generalist models

Addressing this disparity, Wang et al. have introduced a pioneering approach in which images serve as the interface for visual perception. Their models, Painter [78] and its successor SegGPT [79], ingeniously adopt the principles of NLP models, utilizing images both as prompts and outputs. These models leverage three-component tensor inputs: an image paired with its corresponding label (analogous to a support set in FSS literature), and a target image for which a prediction is desired. Similarly, Meng et al. proposed SegIC [48], a model that combines the strengths of foundation models and in-context learning. It employs a pre-trained foundation model for feature extraction, followed by a novel ”correspondence discovery” step. This step establishes semantic and geometric correspondences between the in-context and target feature maps, providing fundamental guidance for segmentation. Subsequently, ”in-context instructions” are extracted based on the in-context samples and these correspondences. Finally, a lightweight decoder leverages this information to generate the segmentation mask for the target image. A key feature of these models and their associated training procedures is their ability to leverage larger pre-training datasets. To this end, to guarantee SegGPT its flexibility, a key factor is its pre-training on a large combination of existing datasets, including ADE20K, MS-COCO [42], PASCAL VOC [18], Cityscapes [14], LIP [41], PACO [56], CHASE_DB1 , DRIVE [21], HRF [8], STARE [30], iSAID [82], and loveDA [76]. Generalist models like SegGPT and SegIC, with their foundations in in-context learning, offer a promising avenue for performing few-shot semantic segmentation on the Carinthia dataset, even with limited labeled data. By formulate the segmentation task as either an image inpainting problem (as in SegGPT, Fig. 8) or a feature correspondence and decoding problem (as in SegIC, Fig. 9), these models can potentially learn to identify and delineate defect regions based on only a handful of example images and their corresponding labels or, ideally, segmentation masks. The inherent ability of these models to generalize from limited examples makes them particularly well-suited to the few-shot learning scenario presented by the Carinthia dataset. Specifically, the adaptation would involve providing the models with a small support set of images, each representing a distinct defect class, and then training them to segment new, unseen images by inferring the segmentation boundaries based on the learned context from the support set. Although implementing these adaptations and achieving optimal performance may necessitate careful model selection, pre-training on large, diverse datasets and fine-tuning, the potential of these generalist models to effectively perform few-shot semantic segmentation on the Carinthia dataset, even without extensive manual mask creation, is significant. Their application could track the way for more efficient and adaptable defect analysis in semiconductor manufacturing.

While foundational and generalist models can be utilized for a wide array of vision tasks, including FSS, there are challenges such as the need for large and diverse training sets, the complexities of training procedures, as well as domain-specific performance variations and the intricacies of parameter tuning. Exploring alternative approaches, such as backbones based on space-state models like MAMBA [24] and

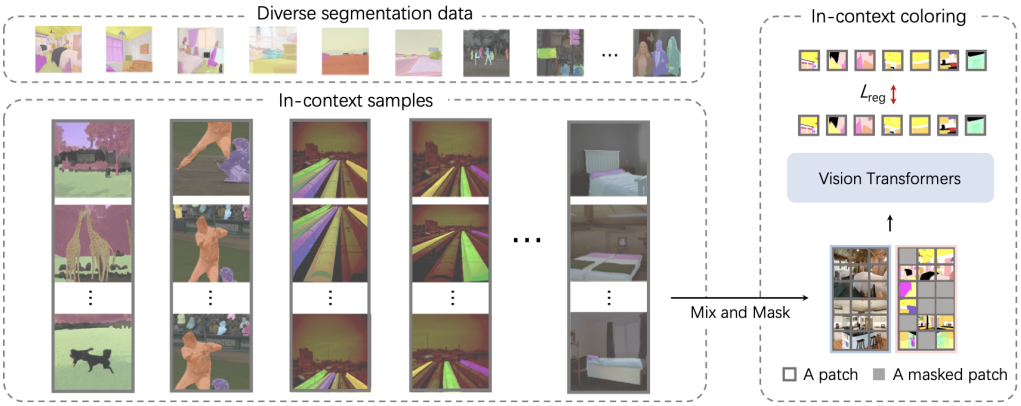


Figure 8. SegGPT unifies diverse segmentation tasks (part, semantic, instance, panoptic, person, medical, aerial) by converting all data into a consistent image format. The model utilizes in-context learning, dynamically generating examples with shared visual contexts. Built upon the Painter framework, SegGPT is trained with an in-context coloring objective and a random coloring scheme for enhanced generalization, taken from [79].

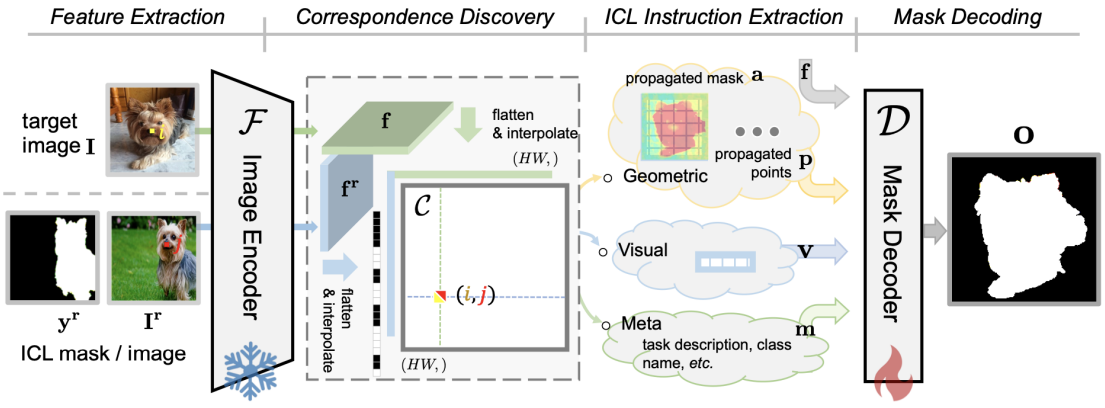


Figure 9. SegIC employs a frozen vision foundation model in a four-stage architecture: feature extraction, correspondence discovery, in-context instruction extraction, and mask decoding. The model first extracts features, then establishes correspondences between them. These correspondences are used to extract instructions that guide the final mask decoding stage, producing the segmentation output. This method utilizes a frozen foundation model for precise segmentation, taken from [48].

VisionMAMBA [94], could potentially offer simplified architectures without compromising performance. In conclusion, the ongoing research and development in these areas promise to further unlock the potential of foundation models.

METRICS COMMONLY USED

Intersection over Union

Object category segmentation performance is typically measured using the Intersection-over-Union (IoU) metric. The IoU assesses the overlap between the predicted region and the ground-truth region of an object within an image. This is calculated by dividing the area of intersection by the area of the union of these two regions.

$$\text{IoU} = \frac{|T \cap P|}{|T \cup P|} \tag{2}$$

Where:

- T: The true label image (e.g., the ground truth bounding box or segmentation mask).
- P: The prediction of the output image (e.g., the predicted bounding box or segmentation mask).
- $T \cap P$: The intersection of sets T and P, representing the overlapping area.
- $T \cup P$: The union of sets T and P, representing the total area covered by both sets.

The IoU value ranges from 0 to 1, where:

- IoU $= 0$: No overlap between T and P.
- IoU $= 1$: Perfect overlap between T and P.

A stands for the true label image T, and P stands for the prediction of the output image. The symbols used in the formula are derived from set theory. The Intersection over Union (IoU) is then calculated as the average over the entire set of pixels, producing an IoU value that ranges between 0 and 1. However, these set symbols are not differentiable. To use these set symbols in their true form, the numbers in T and P must be absolute 1's and 0's. While the label image T inherently contains values of 1's and 0's, the output P contains values between 0 and 1 due to the sigmoid activation in the final up-sampling layer of the neural network. To address this, an approximation of IoU can be made using probabilities. This leads to the following equation for the approximation IoU':

$$IoU' = \frac{|T * P|}{|T + P - (T * P)|} = \frac{I}{U} \tag{3}$$

Where:

- T: Represents the true labels (values from A).
- P: Represents the predicted probabilities (values from B).
- $T * P$: Denotes the intersection, approximated using probabilities.
- $T + P - (T * P)$: Denotes the union, approximated using probabilities.
- I: Simplified to represent the intersection.
- U: Simplified to represent the union.

This approximation allows the IoU metric to be used during training as it becomes differentiable and compatible with optimization algorithms. The sigmoid activation ensures smooth probabilistic predictions, making this approximation viable for gradient-based learning. In the Equation 3, the numerator approximates the intersection of T and P. It essentially calculates the probability of Px only

when the corresponding Tx is 1. If Tx is 0, the contribution to the numerator is 0. This effectively means the intersection is maximized when Px is 1 whenever Tx is 1, reflecting the expected behavior of an intersection.

The denominator calculates the union of T and P, similar to a standard union calculation. It sums the elements of T and P and then subtracts the approximate intersection (T * P). This subtraction prevents double-counting the elements present in both T and P (the intersection), see Equation 3 [5]. Importantly, the IoU effectively handles class imbalance, a frequent challenge in segmentation tasks. For instance, if a naive algorithm classifies every pixel as background, it would be effectively penalized by the IoU, resulting in a score of zero. This is because the intersection between the predicted and ground-truth regions would be zero. While simple loss functions like softmax loss (which optimizes for overall accuracy) are sometimes used, many deep learning segmentation methods employ more specialized loss functions designed to handle class imbalance and other complexities.

Mean Intersection over Union

The Mean IoU is then calculated as the average of the IoU scores across all objects or classes in an image or a dataset. Given n objects or classes, the Mean IoU is given by:

$$\text{Mean IoU} = \frac{1}{n}\sum_{i=1}^{n}\text{IoU}(T_i, P_i) \tag{4}$$

In the Equation 4, P_i is the predicted region for object/class i, T_i is the ground truth region for object/class i, and n is the total number of objects or classes. Mean IoU is valuable because it considers both false positives (incorrectly predicted regions) and false negatives (missed regions). It provides a measure of spatial accuracy in object detection and segmentation. However, Mean IoU can be sensitive to object size, this because larger objects tend to dominate the Mean IoU calculation, potentially masking the performance on smaller objects. Alternative metrics like the F1-score or class-weighted IoU may be considered to address these limitations.

Foreground-Background Intersection over Union

Another used evaluation metric is the Foreground-Background Intersection over Union (FB-IoU). This metric is a specialized variant of the mean Intersection over Union (mIoU), designed to simplify the evaluation process by reducing the problem to a binary classification task. Specifically, FB-IoU treats all foreground objects as a single class and the background as another, effectively reducing the number of categories to two ($C = 2$). Despite its simplicity, FB-IoU has been criticized for several limitations. One major issue is that it ignores the distinct categories of objects within the foreground and this can lead to biased results, particularly in datasets with significant class imbalance, such as MS COCO or PASCAL VOC. For instance, if a dataset contains a large number of small objects and a few large ones, the FB-IoU metric may disproportionately reflect the performance on the larger objects while neglecting the smaller ones. Zhang et al. [86] highlight another critical limitation of FB-IoU: its inability to accurately reflect the performance of a model in segmenting small objects. Since FB-IoU aggregates all foreground objects into a single class, the failure to segment small objects may not significantly impact the overall score. This is particularly problematic in scenarios where the background occupies a large portion of the image; in such cases, the contribution of small objects to the FB-IoU score becomes negligible, even if the model performs poorly on them. Mathematically, FB-IoU can be expressed as:

$$\text{FB-IoU} = \frac{\text{Area of Foreground Intersection} + \text{Area of Background Intersection}}{\text{Area of Foreground Union} + \text{Area of Background Union}} \tag{5}$$

In Equation 5, the foreground and background are treated as two distinct regions, and the metric computes the ratio of the overlapping areas to the union of the areas. However, as discussed earlier, this formulation does not account for the diversity of object categories within the foreground, leading to potential inaccuracies in performance evaluation. Thus, while FB-IoU provides a straightforward and computationally efficient way to evaluate segmentation models, its disregard for individual object categories and its sensitivity to class imbalance limit its effectiveness.

DATASETS

As explained in previous sections, unlike traditional semantic segmentation, which relies on large amounts of annotated data, FSS requires models to generalize to unseen classes with minimal supervision. This problem is particularly relevant in real-world applications where acquiring extensive labeled datasets is impractical.

The PASCAL-5i Dataset

To address the need for a standardized benchmark in FSS, Shaban et al.[63] introduced the PASCAL-5i dataset. This dataset is constructed by simulating segmentation episodes based on the original PASCAL VOC 2012 dataset [18] and its extended annotations from Semantic Boundaries Dataset (SDS) [27]. The PASCAL VOC 2012 dataset contains 20 object classes, which are divided into training and testing subsets for FSS evaluation.

Dataset Construction

Given the set $\mathscr{L}$ of all 20 classes from PASCAL VOC 2012, Shaban et al.[63] sample 5 classes to form the test label set $\mathscr{L}_{test}$, while the remaining 15 classes are used to build the training label set $\mathscr{L}_{train}$. This sampling process is repeated to create 4 distinct folds, ensuring a comprehensive evaluation. Specifically, the test set for each fold i is defined as:

$$\mathscr{L}_{test} = \{4i+1,\ldots,4i+5\}, \quad i \in [0,3]. \tag{6}$$

Once $\mathscr{L}_{test}$ and $\mathscr{L}_{train}$ are defined, the training dataset $\mathscr{D}_{train}$ is constructed by selecting all $(image, mask)$ pairs from the PASCAL VOC training subset that contain objects from $\mathscr{L}_{train}$. Similarly, the test dataset $\mathscr{D}_{test}$ consists of all $(image, mask)$ pairs from the PASCAL VOC validation subset that contain objects from $\mathscr{L}_{test}$. These datasets are then used to sample training and testing episodes, as described in Section 2.1.

Limitations of PASCAL-5i

While PASCAL-5i has been widely adopted as a benchmark for FSS, its limited number of classes (only 20) raises concerns about its suitability for evaluating the generalization capacity of models. Generalization is a critical property in FSS, as models must perform well on unseen classes with minimal training data. The small number of classes in PASCAL-5i may not adequately test a model's ability to generalize across diverse and complex scenarios.

The COCO-20i Dataset

To address the limitations of PASCAL-5i, Nguyen and Todorovic [50] introduced the COCO-20i dataset, a more challenging benchmark based on the MS COCO dataset. Unlike PASCAL-5i, which has 20 classes, COCO-20i employees the 80 object classes in MS COCO. The dataset is constructed using a similar sampling procedure as PASCAL-5i, resulting in 4 folds, each containing 20 classes. This larger number of classes provides a more rigorous test of a model's generalization capabilities. Despite its advantages, COCO-20i has been criticized for its limitations as a benchmark for FSS. Li et al. [39] argue that an ideal FSS benchmark should not only have a large number of classes but also require only a few samples per class for training. COCO-20i, while larger than PASCAL-5i, still falls short in this regard, as it does not fully capture the few-shot learning paradigm.

The FSS-1000 Dataset

To overcome the limitations of both PASCAL-5i and COCO-20i, Li et al. [39] introduced the FSS-1000 dataset, specifically designed for FSS. This dataset contains 1000 classes, each with 10 images, making it significantly larger and more diverse than previous benchmarks. Unlike PASCAL-5i and COCO-20i, which may contain multiple object classes in a single image, FSS-1000 segments only one class per image. This design choice simplifies the evaluation process but also introduces limitations, as it restricts the evaluation to the Foreground-Background Intersection over Union (FB-IoU) metric. FB-IoU treats all foreground objects as a single class, which may not fully capture the performance of models in complex multi-class scenarios as well discussed in the previous subsection.

BENCHMARKS

This section presents a comprehensive collection of benchmark results from key works in Few-Shot Semantic Segmentation (FSS). By synthesizing findings from the literature, the objective aim to provide the reader with a consolidated resource that not only highlights the performance of different models but also offers insights into the evolution of FSS research over time. The tables included here provide valuable insights into the attention received by different benchmark datasets, revealing which datasets are more extensively explored and which remain underutilized.

Performance Trends and Dataset Popularity

The tables presented in this section summarize the performance of various FSS models on these benchmark datasets. By analyzing these results, several trends become evident:

- **PASCAL-5i**: As one of the earliest datasets for FSS, PASCAL-5i has been extensively tested by numerous models. Its well-established evaluation protocol and relatively small size make it an attractive choice for initial model validation. However, its limited number of classes (20) raises concerns about its suitability for evaluating generalization capabilities.
- **COCO-20i**: With its larger scale and diversity, COCO-20i has rapidly gained popularity as a benchmark for FSS. Models evaluated on this dataset are often required to demonstrate stronger generalization capabilities, making it a more rigorous test of performance.
- **FSS-1000**: Despite its potential, FSS-1000 remains less prominent in the literature. This may be due to its recent introduction and the challenges associated with its large scale and single-class segmentation design. However, its unique characteristics make it a valuable resource for future research.

The following table presents benchmark results for Few-Shot Semantic Segmentation (FSS) models across three datasets: PASCAL-5^i, COCO-20^i, and FSS-1000. The models are categorized into specialist models (designed specifically for FSS tasks) and generalist models (capable of handling multiple tasks beyond FSS). The performance is evaluated in both one-shot and few-shot settings, with results reported as mean Intersection over Union (mIoU) scores. Table 1 has been adapted from [9]

Table 1. Few-Shot Semantic Segmentation Benchmark Results

Method	PASCAL-5^i		COCO-20^i		FSS-1000	
	one-shot	few-shot	one-shot	few-shot	one-shot	few-shot
specialist model						
HSNet [49] (ICCV 2021)	67.3	71.6	47.3	55.1	86.7	88.9
DCAMA [64] (ECCV 2022)	69.3	74.9	50.9	58.3	90.1	90.4
VAT [29] (ECCV 2022)	67.9	72.0	41.3	47.9	90.0	90.6
PMNet [10] (WACV 2024)	68.1	73.9	43.7	53.1	84.6	86.3
generalist model						
Painter [78] (CVPR 2023)	64.5	64.6	32.8	32.6	61.7	62.3
SegGPT [79] (ICCV 2023)	83.2	89.8	56.1	67.9	85.6	89.3
SegIC (L) [48] (ECCV 2024)	na	na	76.1	na	86.8	na
SegIC (G) [48] (ECCV 2024)	na	na	74.5	na	88.4	na

Note: The SegIC paper [48] mentions two versions of SegIC, marked as ”L” and ”G”, in the benchmark table. These letters stand for the size of the Vision Foundation Model (VFM) used as the backbone for SEGIC:

- SegIC (L): This version uses DINOv2-L (large) as the backbone.
- SegIC (G): This version uses DINOv2-G (giant) as the backbone.

The difference between these two versions lies in the capacity and performance of the underlying DINOv2 model [52]. DINOv2-G is a larger and more powerful model than DINOv2-L, generally leading to better performance but at the cost of increased computational resources. Essentially, the authors are presenting results using two different "sizes" of their SegIC framework to demonstrate the scalability and effectiveness of their approach with different underlying foundation models. As shown in the table, SegIC(G) generally outperforms SegIC(L) across the different tasks, which is consistent with using a higher capacity backbone.

The results presented in Table 1 provide a comprehensive comparison of various few-shot semantic segmentation methods, including both specialist and generalist models. Among the generalist models, SegGPT and SegIC exhibit notable performance across different datasets, highlighting their relevance for further semiconductor defect segmentation research. SegGPT, presented at ICCV 2023, demonstrates strong performance across all datasets. On the PASCAL-5^1 dataset, SegGPT achieves an impressive one-shot accuracy of 83.2 and a few-shot accuracy of 89.8, significantly outperforming other generalist models such as Painter. Similarly, on the COCO-20^1 dataset, SegGPT attains a one-shot accuracy of 56.1 and a few-shot accuracy of 67.9, which is considerably higher than Painter's performance. In the FSS-1000 dataset, SegGPT maintains robust performance with a one-shot accuracy of 85.6 and a few-shot accuracy of 89.3, indicating its versatility and effectiveness across diverse datasets.

SegIC, introduced at ECCV 2024, also shows promising results, particularly on the COCO-20^1 and FSS-1000 datasets. The SegIC (L) variant achieves a one-shot accuracy of 76.1 on COCO-20^1, which is higher than SegGPT's performance on the same dataset. However, it is important to note that the few-shot results for SegIC are not available (*na*) in the table, which limits direct comparison in that context. On the FSS-1000 dataset, SegIC (L) achieves a one-shot accuracy of 86.8, while SegIC (G) slightly outperforms with a one-shot accuracy of 88.4. These results suggest that SegIC, particularly the (G) variant, may offer competitive performance, especially in scenarios where high one-shot accuracy is crucial. In summary, both SegGPT and SegIC represent state-of-the-art generalist models in few-shot semantic segmentation. SegGPT's consistent performance across multiple datasets makes it a strong candidate for tasks requiring both one-shot and few-shot learning. SegIC, particularly the (G) variant, shows potential for achieving high one-shot accuracy, which could be advantageous in specific applications. Further empirical testing of these models could provide deeper insights into their relative strengths and applicability to the semiconductor defects specific use case.

CHALLENGES AND LIMITATIONS

The fast progress in few-shot semantic segmentation (FSS) has also brought ongoing issues that slow down its use in real-world applications; for example, one fundamental limitation lies in the reliance on unrealistic assumptions during model training and testing phases. Many current methods expect to know new classes during training, like removing images with unknown categories from the basic training set to falsely improve results [26]. This approach helps in controlled tests but fails in real situations where new classes are completely unknown when training. This mismatch between experimental protocols and practical conditions leads to overestimated generalization capabilities. Also, most methods need exact matches between support and query sets, requiring researchers to choose support samples that clearly show the target categories present in the query images [26]; this manual process needs continuous adaptation to changing environments. Another problem comes from the inability to capture detailed object features using whole-object summary representations. Traditional prototype-based methods generate single vector representations for entire classes, which struggle to account for intra-class variations caused by differences in object parts, poses, or subcategories [44]. For instance, when segmenting animal categories with diverse appearances, a holistic prototype may average out distinctive features like limb positions or texture patterns, leading to incomplete coverage of target regions. This problem exacerbates in semi-supervised settings with limited labeled data, where coarse prototypes fail to leverage unlabeled examples effectively to model nuanced feature distributions. This issue gets worse in semi-supervised learning with limited labeled data, where coarse summaries are ineffective in utilizing unlabeled examples to accurately represent detailed feature patterns [44]. Recent attempts to decompose prototypes into part-aware components show promise but introduce computational complexity in clustering and refining these sub-prototypes through attention mechanisms, making them harder to use in real-time. Domain shift and dataset bias further compound these challenges, particularly when deploying models in environments with sensory or contextual differences from training data. While FSS models demonstrate reasonable generalization within similar domains,

their performance degrades significantly when test data exhibits substantial distributional shifts in lighting conditions, object scales, or background contexts [4]. This limit is clear in robotics and remote sensing applications, where deployment environments often differ radically from curated benchmark datasets [22]. Current mitigation strategies like meta-learning have limited success because they usually focus on narrow task types instead of creating strong adaptability across different data types. The computational complexity of multi-stage architectures presents another barrier to practical implementation. Many state-of-the-art methods use complex systems with separate networks for summary creation, feature matching, and mask refinement. For example, approaches combining graph attention networks with iterative prototype refinement achieve improved accuracy but require extensive computational resources for both training and inference [44]; this architectural overhead contradicts the core motivation of few-shot learning—enabling efficient adaptation with minimal resources. Also, the close connection between different pipeline components makes these systems vulnerable to component-level failures, as errors in prototype generation propagate irreversibly through subsequent stages [26]. A critical yet under-addressed challenge involves maintaining base-class performance while adapting to novel categories in generalized few-shot segmentation (GFSS) scenarios. Current methods exhibit significant performance degradation on base classes after adapting to new ones, especially when dealing with many new categories at once and this phenomenon occurs because adaptation mechanisms primarily focus on novel class features while inadvertently distorting the feature space representations of base classes. The problem intensifies in class-imbalanced scenarios where new classes appear infrequently compared to existing base classes. Recent studies show that even advanced GFSS methods have trouble getting equal performance between base and new classes, with average accuracy differences over 15% in standard tests [62]. The limited scalability to multi-way segmentation tasks restricts real-world applicability. In fact, most FSS methods focus on binary segmentation (separating single novel class from background), while practical scenarios require simultaneous identification of multiple novel and base classes. Expanding current architectures to handle multi-way segmentation introduces exponential complexity in prototype matching and mask generation; some attempts to address this through parallel prototype branches or hierarchical matching strategies show modest improvements but substantially increase memory footprint and inference latency [44]. Additionally, the lack of standardized evaluation protocols for multi-way few-shot segmentation makes comparative analysis challenging across different implementations. These limitations collectively highlight the need for fundamental architectural innovations rather than incremental improvements to existing paradigms. The path forward likely requires rethinking feature representation learning, developing more efficient adaptation mechanisms, and establishing rigorous evaluation frameworks that better reflect real-world operational conditions [75].

APPLICATION OF A GENERALIST BASED MODEL FOR THE SPECIALIZED INDUSTRIAL DOMAIN

Reasons for employing a Transformer-Based model for Semiconductor Defect Segmentation

The selection of a Transformer-based model, specifically SegGPT, for the segmentation of defects in semiconductor images is based on a confluence of factors that render this approach particularly advantageous compared to traditional segmentation methodologies. Semiconductor defect segmentation presents unique challenges that necessitate sophisticated and adaptable solutions, and models like SegGPT offer based on the literature benchmarks capabilities to address these complexities effectively.

Traditional approaches to semiconductor defect segmentation, often relying on Convolutional Neural Networks (CNNs) trained for specific defect types or rule-based algorithms, encounter limitations when confronted with the variability and evolving nature of defects in semiconductor manufacturing. These challenges include:

- **High Variability in Defect Appearance:** Semiconductor defects manifest in diverse forms, sizes, and appearances. Factors such as lighting conditions, imaging modalities (e.g., SEM, optical microscopy), and the specific layer of the semiconductor wafer being imaged contribute to significant intra-class variability. Traditional CNNs, trained on limited datasets of specific defect types, often struggle to generalize to the full spectrum of defect manifestations encountered in real-world production environments.

- **Contextual Importance in Defect Identification:** Identifying defects accurately is not solely dependent on pixel-level characteristics but also on the broader contextual information within the semiconductor image. The location of a potential defect relative to circuit patterns, repetitive structures, and other features on the chip is crucial for distinguishing genuine defects from benign variations or noise. Transformer architectures, inherent to generalist models like SegGPT, excel at capturing long-range dependencies and contextual relationships within images, enabling a more holistic understanding of the defect within its surrounding environment.

- **Need for Generalization and Reduced Data Dependency:** Training robust CNN-based segmentation models for each specific defect type requires substantial amounts of labeled data, which is often expensive and time-consuming to acquire in semiconductor manufacturing. generalist models, pre-trained on vast datasets and designed for general-purpose vision tasks, demonstrate superior generalization capabilities. SegGPT, specifically, leverages a generalist approach to segmentation, allowing it to adapt to new segmentation tasks with fewer task-specific training examples. This potential for reduced data dependency is a significant advantage in the context of semiconductor defect analysis, where labeled defect datasets can be limited.

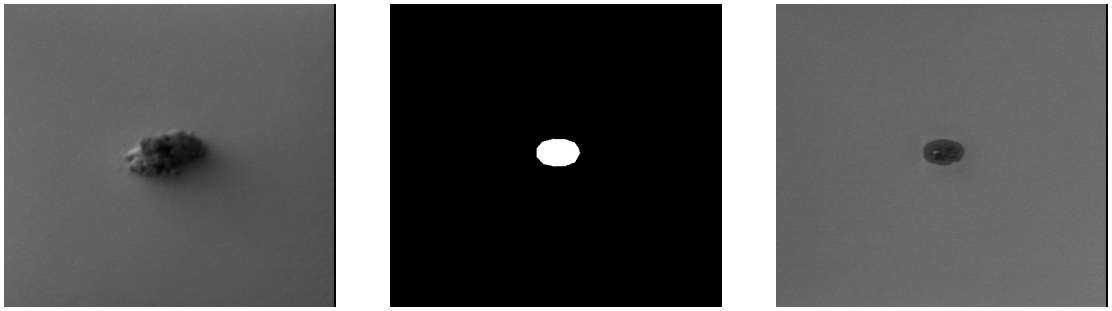
Transformer-based models, and SegGPT in particular, offer a effective alternative to traditional methods for semiconductor defect segmentation due to the various key advantages. For example, the self-attention mechanism inside the Transformer architectures allows the model to effectively capture long-range dependencies and contextual information within semiconductor images. This enables SegGPT to understand the relationship between potential defects and their surrounding features on the chip, leading to more accurate and context-aware segmentation. SegGPT's in-context learning paradigm allows it to adapt to new segmentation tasks and even novel defect types by conditioning on prompt images and masks. This flexibility significantly reduces the need for extensive retraining and re-annotation when encountering new defect morphologies, making it highly adaptable to the evolving landscape of semiconductor manufacturing. Pre-trained generalist models like SegGPT demonstrate superior generalization capabilities compared to defect-specific CNNs. Their training on large and diverse datasets equips them with more robust feature representations, enabling them to handle the high variability in defect appearance and imaging conditions characteristic of semiconductor defect images. Basically, the generalist nature of these types of models and their in-context learning capabilities suggest a potential for reduced data dependency for adapting to specific semiconductor defect segmentation tasks. This is particularly advantageous in scenarios where labeled defect data is scarce or expensive to obtain. In conclusion, the adoption of a Transformer-based generalist model, such as SegGPT, for semiconductor defect segmentation is justified by its inherent capabilities to address the unique challenges of this domain. Its ability to capture contextual information, adapt to novel defects through in-context learning, generalize across diverse defect appearances, and potentially reduce data requirements makes it a powerful and promising approach for advancing the accuracy, efficiency, and adaptability of semiconductor defect detection and segmentation in modern manufacturing processes. The flexibility and robustness offered by SegGPT represent a significant step forward compared to traditional, more specialized segmentation techniques in the context of complex and evolving semiconductor defect analysis.

IMPLEMENTATION DETAILS

This section details the implementation of the SegGPT model in python code. SegGPT, as introduced in [79], is a robust decoder-only Transformer model designed for versatile image segmentation. The implementation utilizes the Hugging Face `transformers` library, which provides pre-trained models and utilities for diverse transformer-based architectures, including SegGPT. The implementation process comprises the following key steps:

Environment Setup and Library Imports

The initial step involved establishing the Python environment and importing the requisite libraries. Note: Since the GPU on my machine is not manufactured by NVIDIA, Google Colab with GPU enabled runtime was utilized for this implementation. This ensures that all necessary tools are available for subsequent phases. This primarily involved importing libraries for tensor operations, model handling, image manipulation, file path management, visualization, and numerical computation.



(a) Input_image **(b)** Prompt_mask **(c)** Prompt_image

Figure 10. A comparison of the approach proposed to quantitative assess the performance of the model

Input Image and Prompt Preparation

This phase focused on loading and preparing the input image intended for segmentation, along with the prompt image and mask to guide the segmentation. File paths were defined for the input image, prompt image, and prompt mask. Subsequently, these images were loaded using the Pillow library. Here, the prompt mask was converted to grayscale format, while the input and prompt images were maintained in RGB format.

Model and Image Processor Loading

The pre-trained SegGPT model and its associated image processor were loaded using the `from_pretrained()` method provided by the `transformers` library. The checkpoint `"BAAI/seggpt-vit-large"` was specified, representing a SegGPT model pre-trained on an extensive dataset and recommended for general applications.

The loading of the `SegGptImageProcessor` is a fundamental part, as it manages the essential preprocessing steps to prepare the input images and masks for the model; this includes resizing, normalization, and conversion to the tensor format. `SegGptForImageSegmentation` is the model class that includes the SegGPT architecture in conjunction with a segmentation head, making it suitable for direct image segmentation tasks. The model requires three input images to perform segmentation using a 1-shot learning approach: the input_image (Fig. 16a), the prompt_image (Fig. 10b), and the prompt_mask (Fig. 10c). The input_image is the target image for which the segmentation mask is generated. The prompt_image serves as the reference image containing the defect, acting as the ground truth. The prompt_mask is the corresponding segmentation mask of the prompt_image, providing the model with the defect's precise location.

Input Preprocessing

The `SegGptImageProcessor` was employed to preprocess the input image, prompt image, and prompt mask. This step transforms the PIL images into the format expected by the SegGPT model, specifically PyTorch tensors, and applies necessary transformations such as resizing and normalization. The parameter `return_tensors="pt"` ensured that the output was in PyTorch tensor format.

Inference Execution

Inference was executed using the loaded SegGPT model. The `torch.no_grad()` context manager was utilized to disable gradient calculations during inference, thereby accelerating the process and reducing memory footprint; the preprocessed inputs were then passed to the model. This step effectively performs the forward pass through the SegGPT network to produce segmentation predictions.

Post-processing for Semantic Segmentation

The raw outputs from the model were then post-processed using the `image_processor.post_process_semantic_segmentation()` method. This stage converts the model's output logits into semantic segmentation masks. `target_sizes` was set to the original dimensions of the input image to ensure alignment of the segmentation mask with the input image dimensions.

Visualization of Results

To visually evaluate the segmentation outcomes, `matplotlib.pyplot` was used to display the input image alongside the generated segmentation mask in a side-by-side subplot configuration. The images were converted to NumPy arrays to ensure compatibility with `matplotlib` for display purposes. This visual inspection allows for a qualitative assessment of the segmentation performance.

RESULTS AND OBSERVATIONS

Upon execution, the implementation successfully loaded the pre-trained SegGPT model and processed the input image using the provided prompt. A segmentation mask was generated and visualized adjacent to the input image.

The shape of the predicted mask tensor was printed to the console, for instance, displaying a shape like `torch.Size([178, 297])` (the exact shape varies depending on the input image size).

This shape reflects the height and width of the segmentation mask, consistent with the resized dimensions of the input image post-preprocessing. Visually, the segmentation mask effectively highlights the foreground object in the input image, this is basically the defect in the semiconductor. The segmentation appears to be reasonably accurate (Fig. 11).

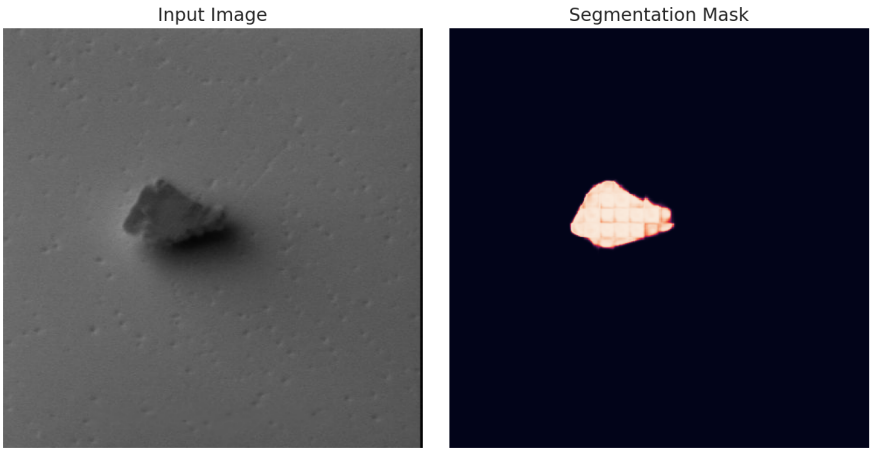


Figure 11. Comparison between the original mask and the predicted mask with SegGPT model (large) with 1-shot learning approach.

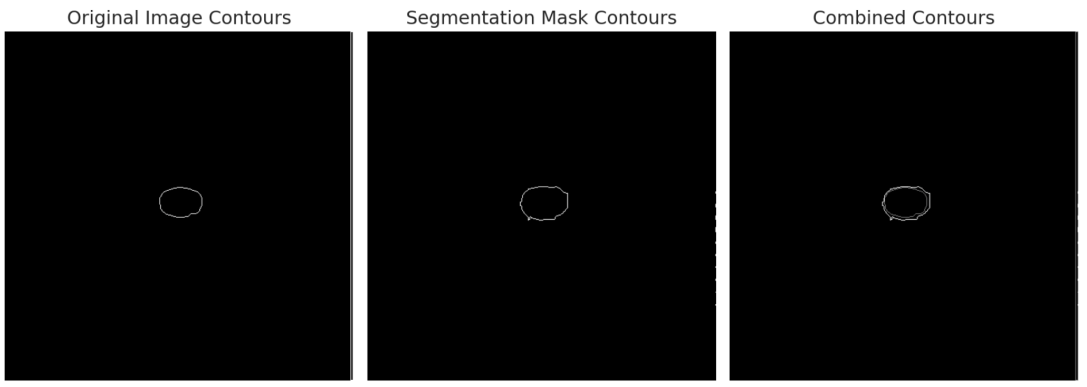


Figure 12. Comparison between the contours derived from the orignal images and the segmentation mask generated from SegGPT model (large).

Possible quantitative model evaluation for 1-shot learning

Given the inherent challenge of evaluating one-shot semantic segmentation in the complete absence of ground truth labels (a single mask has been created for model prompting), has been devised a quantitative evaluation approach based on the principle of comparing the structural features of the predicted segmentation masks with those present in the original input images. This approach takes the idea that, while we don't know the true semantic segmentation, a reasonable prediction should, at the very least, exhibit boundaries that align with prominent edges and intensity changes within the original image. This is, of course, an indirect measure and relies on certain assumptions, which will be discussed later. The core of the following method involves extracting contours, which represent the boundaries of objects or regions, from both the original image and the predicted segmentation mask, and then calculating metrics that quantify the degree of agreement between these two sets of contours.

For the original image, preprocessing steps are applied to enhance edges and create a binary representation suitable for contour extraction, in particular, the image is converted to grayscale, followed by Gaussian blurring to reduce noise. Otsu's thresholding method is then used to automatically generate a binary image, differentiating foreground (potential defect regions) from background. Morphological dilation is performed to close small gaps and consolidate detected regions and subsequently, the 'findContours' function from the OpenCV library identifies contours in this processed binary image. A filtering step is applied to mitigate spurious contours arising from noise, retaining only contours enclosing an area exceeding a predefined threshold; this filtered set of contours represents the derived "pseudo-ground truth" based on the image's structural information. For the predicted segmentation mask, which is assumed to be a binary image representing the model's output, the process is streamlined. The 'findContours' function is directly applied to extract boundaries of predicted defect regions, without requiring preprocessing. Contours are then extracted from both the processed original image and the predicted mask, enabling the calculation of two key metrics: average Euclidean distance and overlap percentage; a visualization of this approach is in Fig. 12. The average Euclidean distance provides a measure of spatial proximity between the contours. Contour sets are first converted into sets of (x, y) coordinate points. Pairwise Euclidean distances are computed between all points on the predicted mask's contours and all points on the original image's contours. The minimum distance from each point on the predicted contour to the original image contour is determined, and the average of these minimum distances constitutes the final metric. A lower average distance suggests closer alignment between predicted mask boundaries and prominent edges in the original image. The overlap percentage quantifies the extent to which the area enclosed by the predicted mask's contours coincides with the area enclosed by the original image's contours. Binary masks representing the filled areas defined by the contours are created. Overlap is calculated as the intersection area of these two masks, divided by the area of the predicted mask, expressed as a percentage. Normalization by the predicted area increases sensitivity to under-segmentation, but conversely, can be artificially inflated by over-segmentation. A high overlap does not definitively guarantee accurate segmentation; it indicates that the predicted region encompasses a substantial portion of features detected in the original image's processed representation. The metrics explained previously and the results in Table 2, in conjunction with visual inspection, offer a constrained, yet quantifiable, means of assessing

Table 2. Summary of Quantitative Evaluation Metrics applied to fig.12

Metric	Value
Average Euclidean Distance (pixels)	2.59
Overlap Percentage (%)	16.59

model performance without access to true labels. Interpretation of these metrics necessitates caution, acknowledging the inherent limitations of this indirect evaluation approach.

Comparison with SAM2 model

This research also involves contrasting SegGPT with the Segment Anything Model 2 (SAM2), Meta's second version of the Segment Anything Model (SAM). While both SegGPT and SAM2 are powerful, general-purpose segmentation models employing the Transformer architecture, they have key architectural and philosophical differences that make a direct comparison highly relevant to understanding the strengths and weaknesses of the chosen approach (SegGPT) for semiconductor defect segmentation. This comparison will explain where SegGPT excels and where SAM2 might offer advantages or different approaches. In this research has already been previously discussed the models architecture in detail, in the methodology review section. Architecturally, SegGPT and SAM2, although both use the Transformer, differ significantly. SegGPT adopts an in-context learning approach, treating different segmentation tasks (including different object classes) as distinct "colors" within a unified segmentation framework. It relies heavily on visual prompts – images and masks provided at inference time – to guide the segmentation. This makes it fundamentally a few-shot or even zero-shot learner, designed to adapt to new tasks with minimal or no task-specific training. The encoder combines the input image with a prompt image (if provided) or a learnable prompt embedding and the decoder outputs segmentations based on the prompt context.

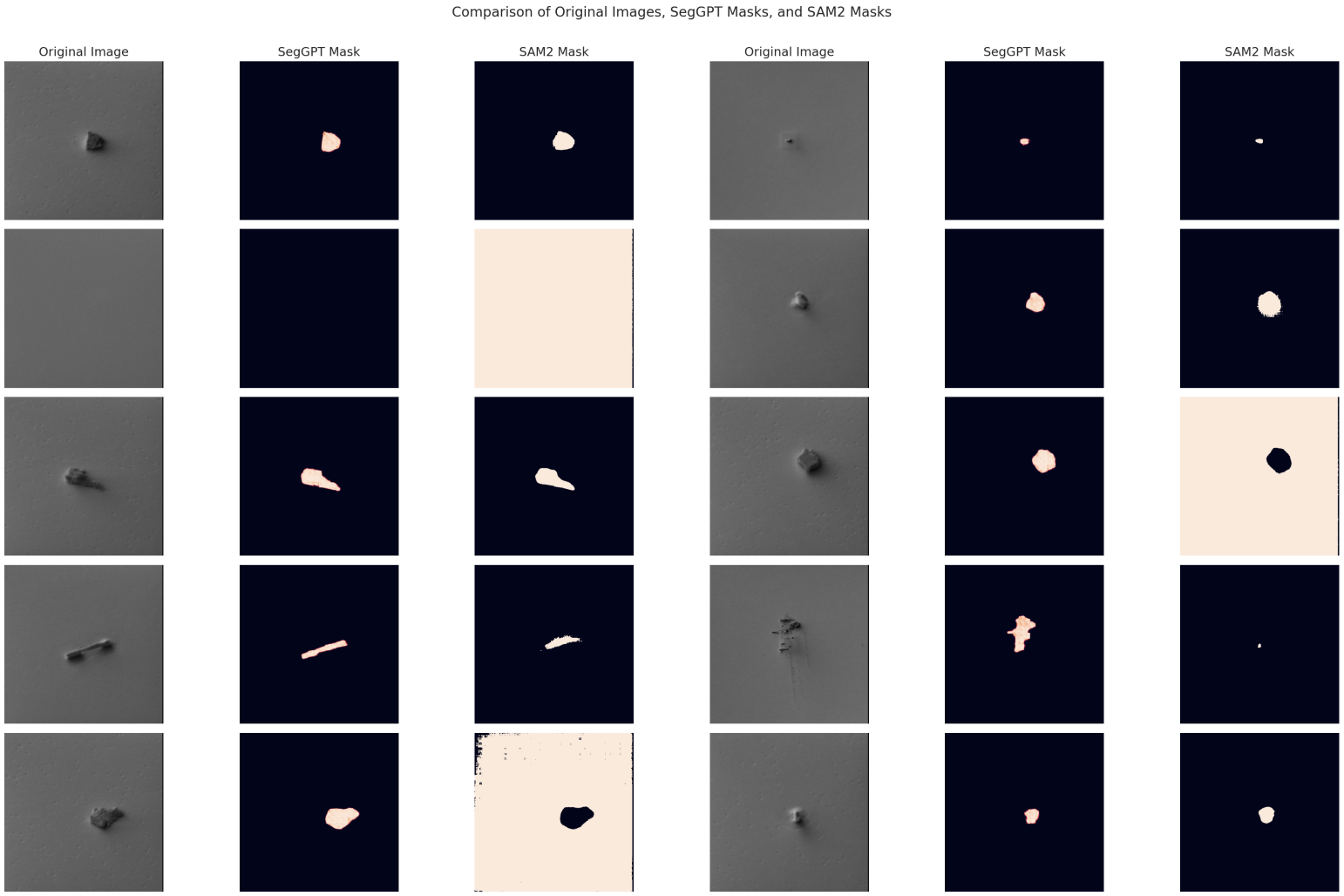


Figure 13. Comparison between the mask generated from SAM2 and segGPT model (large, 1-shot learning approach) on 10 random images

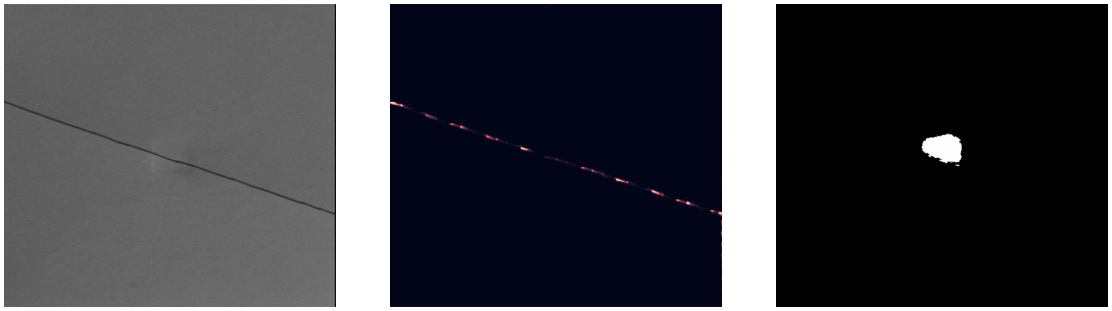
SAM2, primarily relies on prompt engineering but in a different manner. It accepts sparse prompts (points, boxes) and dense prompts (masks) to specify the region of interest. It uses a two-stage architecture: a heavy image encoder and a lightweight prompt encoder and mask decoder. The image encoder computes a fixed image embedding, and the prompt encoder embeds the user-provided prompts; these embeddings are then combined in the mask decoder to produce segmentation masks. To be noted, while SAM2 can handle various prompt types, it's primarily trained on a vast dataset of masks, making it a more "data-hungry" model than SegGPT's in-context learning paradigm suggests. To conlcude, SAM2 is trained on a vast 1-billion mask dataset (SA-1B), making a comparison between a large and small dataset. In the implementation, the segmentation process is carried out using the SAM2 model available on Hugging Face, which inherently requires input prompts in the form of coordinates that delineate the bounding box around defects. To obtain these coordinates, a Visual Language Model (VLM) has been integrated into the workflow. Specifically, the Gemini API, utilizing the gemini-1.5-pro model, is employed to detect and track the bounding box around each defect. The detected coordinates are then converted into JSON format and supplied as the input prompt to the SAM2 model, thereby facilitating accurate defect segmentation. This integration of the Gemini API with SAM2 effectively bridges the gap between defect localization and segmentation by using the advanced capabilities of VLMs to generate the necessary input prompts. However, while this approach has proven effective, it relies on external prompt generation, which introduces an additional step in the processing pipeline. A prospective future direction involves the implementation of an automatic mask generation approach that circumvents the need for manual input prompts. Instead of using the Hugging Face interface, this method would involve directly downloading and utilizing the SAM2 model checkpoints and configuration files. By building the SAM model in accordance with the official documentation, it would be possible to employ an end-to-end segmentation framework that autonomously generates segmentation masks. Although this alternative approach is inherently more complex, as it requires a deeper integration of the model's configuration and checkpoint management, it promises a more streamlined and potentially robust segmentation process. This future strategy may enhance the automation of the segmentation pipeline and reduce reliance on external modules for prompt generation, opening a possible way for more efficient and autonomous defect detection systems. Essentially, SegGPT in conducted experiments show superior performance when compared with SAM2 (Fig. 13).

Failure cases

While a comprehensive quantitative evaluation across the entire dataset of over 4000 images is computationally intensive for a model-by-model, image-by-image visual inspection, a qualitative analysis focusing on representative examples and identified edge cases reveals significant differences in the segmentation performance of SegGPT and SAM2. Specifically, in-depth examination of segmentation outputs revealed instances where SAM2 exhibited difficulty in accurately delineating the defect mask, particularly for certain images associated with a specific defect label (Label 2), as illustrated in Fig. 14b and 14c . These images presented challenges that SAM2, despite its generally strong performance, struggled to overcome. The precise reasons for these failures in SAM2 are likely multifaceted, potentially relating to subtle variations in defect appearance, imaging conditions, or contextual features within the Label 2 images that deviate from the patterns learned during its training on the SA-1B dataset. Although trained with billion of masks, the pre-training, while extensive, might not fully encompass the specific characteristics of these edge cases. Another failure case, but in this case in both models, is highlighted in the Fig. 15b and 15c. A critical finding from this analysis is SegGPT's successful segmentation of Label 2 edge case images, specifically when employing prompt masks and prompt images that closely resemble the input image. In such configurations, mirroring the visual characteristics of the target image in the prompts, SegGPT effectively overcomes the segmentation challenges associated with these edge cases, outperforming models like SAM2 (another comparison also in Fig. 16b and 16c). This underscores the power of SegGPT's in-context learning approach, which excels with visually prompts, especially for nuanced, underrepresented data like Label 2 edge cases.

POTENTIAL APPLICATIONS IN RESEARCH

The successful implementation of SegGPT unlocks several research areas in the FSS Semantic segmentation tasks applied to industrial domain images. SegGPT's in-context learning and generalist nature render it particularly valuable for few-shot segmentation, interactive segmentation, domain adaptation



(a) Input_image **(b)** SegGPT **(c)** SAM2

Figure 14. A comparison of different segmentation mask applied to the same image: (a) Original image, (b) Semantic Segmentation mask from SegGPT (c) Semantic Segmentation mask from SAM2

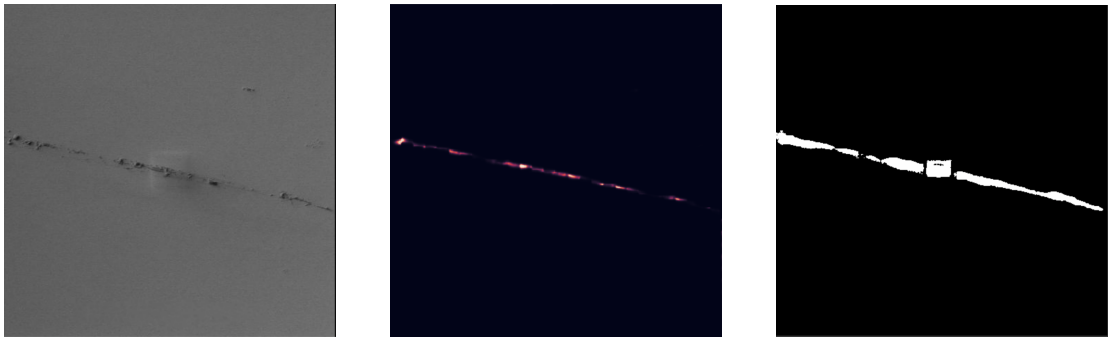
and generalization, and novel segmentation tasks. In scenarios with limited annotated data, SegGPT can be effectively employed for segmenting new objects or categories by providing only a few prompt examples. SegGPT's prompt-based approach also facilitates interactive segmentation workflows, allowing researchers to iteratively refine segmentation masks by providing varied or refined prompts, making it a valuable tool for tasks necessitating human-in-the-loop segmentation, such as automatic detection and segmentation of defects in industrial domann images. Furthermore, SegGPT's ability to generalize across different segmentation tasks and datasets makes it promising for domain adaptation research. It can be utilized to adapt segmentation models trained on one domain to new, unseen domains with minimal fine-tuning, crucial for deploying models in real-world scenarios where data distributions can shift.

Limitations and Future Work

While SegGPT demonstrates significant potential, certain limitations and future research directions warrant consideration. Computational cost is a limitation, as SegGPT, being a transformer-based model, can be computationally demanding, particularly for high-resolution images. Optimizations for inference speed and memory efficiency may be necessary for real-time or resource-constrained applications. Prompt sensitivity is another consideration; segmentation quality is influenced by the selection of prompt images and masks, and investigating strategies for optimal prompt selection and generation could further enhance performance and robustness. Generalization to unseen tasks also requires further attention. While designed for generalizability, SegGPT's performance on entirely novel and significantly different segmentation tasks requires further evaluation, and research into expanding its in-context learning capabilities to handle a broader task spectrum is warranted. Quantitative evaluation is fundamental for rigorous research. Quantitative evaluation of SegGPT's performance on relevant datasets using standard segmentation metrics (e.g., IoU, Dice score) is needed to provide a more objective assessment of its effectiveness compared to alternative segmentation methods. This implementation successfully demonstrates the utilization of the pre-trained SegGPT model for image segmentation. The results are promising (Fig. 17), underscoring SegGPT's potential as a versatile and powerful tool for diverse segmentation tasks within research. Its in-context learning capability and generalist nature offer substantial advantages for research scenarios involving limited data, interactive workflows, domain adaptation, and novel segmentation challenges. Continued research and development, as outlined in the limitations and future work section, can further unlock SegGPT's full potential for advancing research in few shot learning semnatic segmentation applied to industrial domain images. A key observation, derived from the qualitative analysis, is that SegGPT's performance is significantly enhanced when the prompt mask and prompt image have visual similarity to the input image being segmented, especially for some problematic edge cases. This suggests that providing prompts that are contextually relevant and visually analogous to the target instance allows SegGPT to effectively leverage its in-context learning capabilities and adapt to subtle variations in the data distribution. Building upon this observation, a promising direction for future research lies in systematically exploring the impact of increasing the number of prompt examples provided to SegGPT. While the current analysis focused on a 1-shot learning scenario, exploring the inherent in-context learning paradigm of SegGPT further by incorporating multiple (e.g., two or more) prompt images and corresponding masks could potentially lead to even greater generalization capabilities. The rationale behind this is that by

(a) Input_image **(b)** SegGPT **(c)** SAM2

Figure 15. A comparison of different segmentation mask applied to the same image: (a) Original image, (b) Semantic Segmentation mask from SegGPT (c) Semantic Segmentation mask from SAM2



(a) Input_image **(b)** SegGPT **(c)** SAM2

Figure 16. A comparison of different segmentation mask applied to the same image: (a) Original image, (b) Semantic Segmentation mask from SegGPT (c) Semantic Segmentation mask from SAM2

providing a richer and more diverse set of visual examples that are still relevant to the target domain, the model can better capture the underlying features and variations of the defects. For instance, instead of a single edge case example from Label 2 and another from a different label, providing multiple diverse examples from Label 2 itself, or even a combination of edge cases and more typical examples from the same label, might equip SegGPT with a more comprehensive understanding of the target defect characteristics. It is important to consider why the current SegGPT model, as described in the text, might be implicitly or explicitly limited to a 1-shot, fixed number of prompts. Many current models, especially those designed for few-shot learning, often have architectures tailored to a specific number of input prompts due to design choices made for computational efficiency, training stability, or simplicity in implementation. For example, the model's input layers might be configured to accept a fixed number of prompt image and mask pairs, and the subsequent feature fusion or attention mechanisms might be designed to operate on this fixed set of inputs. Without modifying the model architecture, directly feeding more than a predefined number of prompts might not be possible or could lead to unexpected behavior. To enable SegGPT or similar models to effectively utilize more than one prompt images and masks, architectural modifications would likely be necessary. One potential approach is to move away from fixed-size input layers for prompts and towards more flexible mechanisms. For instance, instead of concatenating feature representations of a fixed number of prompts, one could employ techniques like feature aggregation or pooling to combine information from a variable number of prompts. Attention mechanisms could also be adapted to process a sequence of prompt representations, allowing the model to understand the importance of each prompt dynamically. Exploring these architectural modifications will be essential to unlock the full potential of visual prompting for segmentation and create models that can effectively work with a larger set of visual cues for improved generalization, especially in complex and vast scenarios like defect detection.

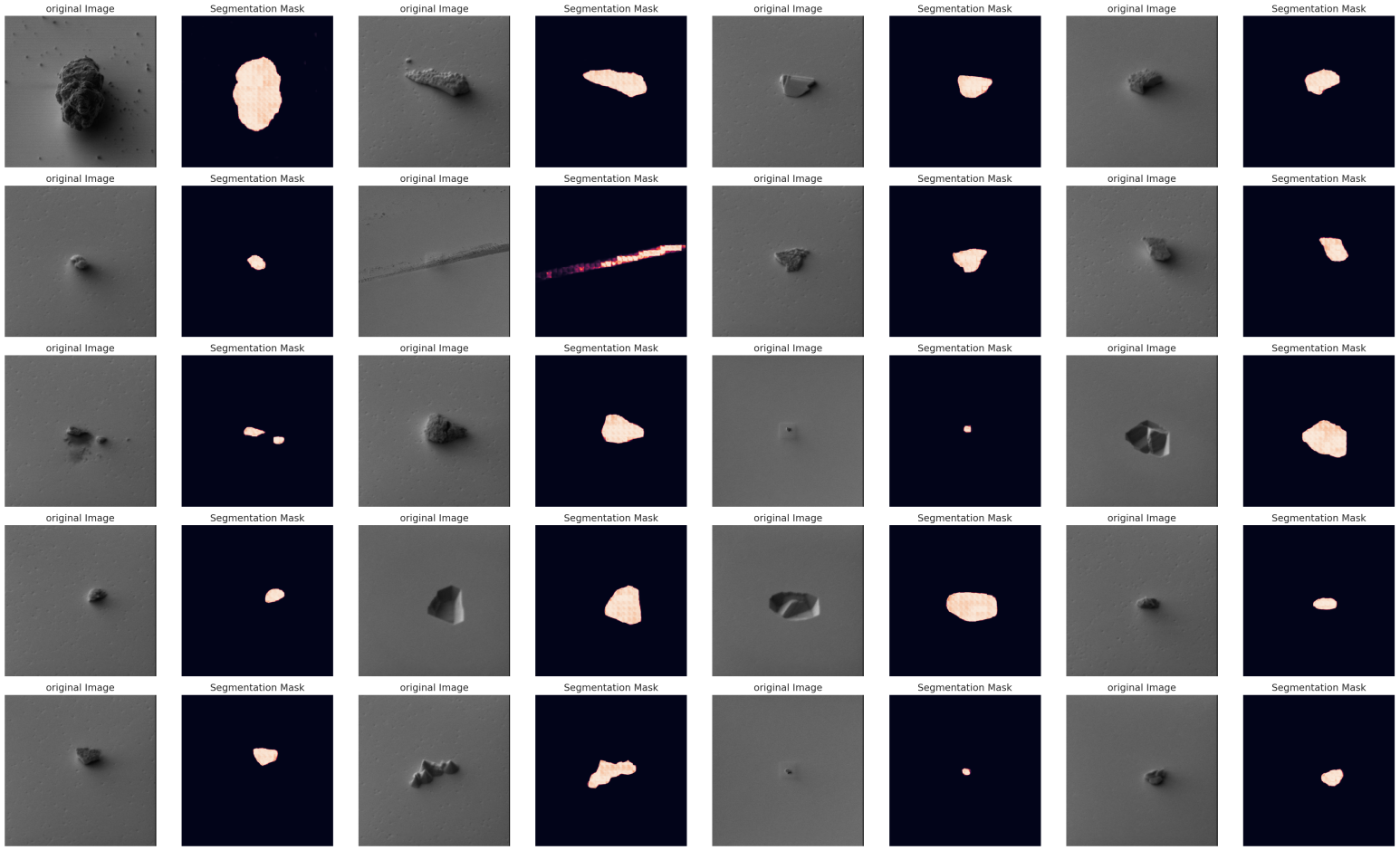


Figure 17. Comparison between the original image and the predicted mask with segGPT model (large) with one shot learning approach on 20 random images

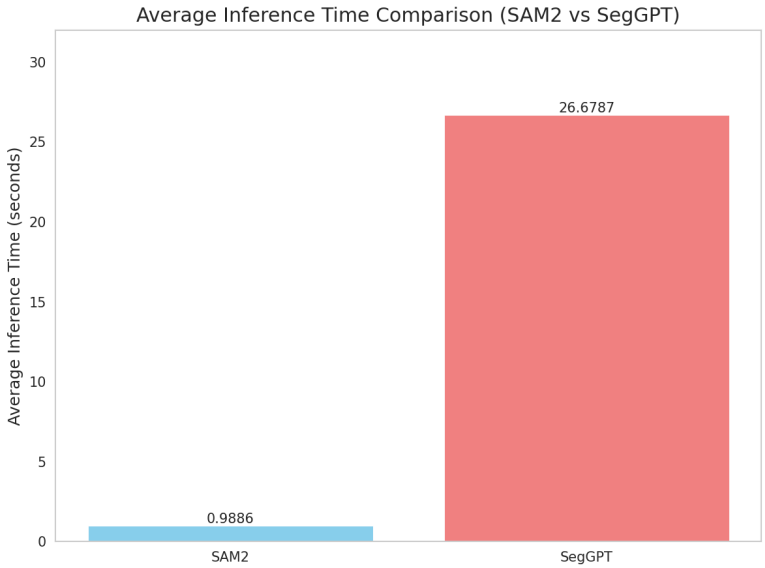


Figure 18. Benchmark between the mask generation time from SAM2 and segGPT model (large, 1-shot learning approach) on 10 random images (average)

CONCLUSION

The exploration and subsequent implementation of models such SegGPT and SAM2 for the specific task of semiconductor defect segmentation constitutes a significant and promising result, giving a robust proof of concept that underscores the potential of generalist vision models to address highly specialized challenges within industrial domains. This work successfully crossed the complexities of adapting a cutting-edge, pre-trained model, originally designed for broad applicability, to the requirements of defect detection in semiconductor manufacturing. The successful implementation, employing the accessible Hugging Face Transformers library, not only demonstrates the practical feasibility of using such advanced architectures but also provides compelling qualitative evidence of SegGPT's capability to perform effective segmentation of defects in semiconductor field (Fig. 17). These initial results, achieved with a one-shot learning approach, based on a paradigm away from traditional, data-hungry methods towards more adaptable and efficient strategies for industrial image analysis. The inherent in-context learning mechanism of SegGPT appears particularly well-suited to the realities of defect analysis, where the vast diversity of defect morphologies, coupled with the often limited availability of meticulously labeled training data, presents a formidable obstacle to conventional supervised learning techniques. A performance benchmark has been conducted between SegGPT and SAM2 model, the results provide a clear comparison of inference efficiency between SAM2 and SegGPT (Fig. 18). When processing a batch of ten image masks, SAM2 demonstrates significantly lower inference times, taking approximately 9.88 seconds to complete the segmentation task, whereas SegGPT requires around 266.78 seconds for the same set of images. This substantial difference in computational speed highlights a crucial trade-off between efficiency and segmentation performance. Despite its slower inference time, SegGPT likely provides a higher level of segmentation accuracy, making it a strong candidate for tasks that demand precision in defect detection. On the other hand, SAM2's much faster processing speed makes it an appealing option for real-time industrial applications where rapid defect identification is critical. The results indicate that SAM2 is well-suited for scenarios where efficiency and responsiveness are the primary concerns, while SegGPT remains a more suitable choice for applications where achieving the highest possible segmentation accuracy is a priority. This trade-off suggests that selecting the most appropriate model depends largely on the specific constraints and requirements of the industrial setting, particularly in semiconductor manufacturing, where both processing speed and segmentation reliability are crucial considerations. However, in meticulously analyzing the performance characteristics of SegGPT within this specialized context, certain critical observations and limitations have emerged, providing valuable insights for future research and development. A key finding is the observable sensitivity of SegGPT's

segmentation accuracy to the nature and quality of the provided prompt images and masks. While the model demonstrably excels when presented with prompts that exhibit strong visual resemblance to the input image undergoing segmentation – particularly in challenging edge case scenarios characterized by subtle or atypical defect appearances – its performance can become less consistent when the prompts are less representative. This behavior highlights a critical trade-off: while carefully selected, visually similar prompts can unlock remarkable segmentation fidelity, especially for complex defect types, a reliance on such specific prompts may accidentally compromise the model's generalization capacity across the full spectrum of defect variations encountered in real-world semiconductor manufacturing environments. Furthermore, the investigation uncovered that optimizing prompt selection for edge cases with similar prompts might unintentionally reduce performance on more typical, non-edge case images, and vice versa. This delicate balance necessitates careful consideration of prompt strategy and suggests that a single, universally optimal prompt may not exist for all defect types or imaging conditions. To address this inherent prompt sensitivity and to enhance the overall robustness and generalization capabilities of SegGPT in the context of semiconductor defect segmentation, the exploration of multi-prompt learning emerges as a particularly promising way for future research. The idea behind this approach is rooted in the understanding that providing a richer, more diverse, and yet still contextually relevant set of prompt examples could empower the model to better capture the underlying statistical distribution of defect characteristics. By moving beyond a single prompt example and instead feeding the model with multiple prompt images and their corresponding masks, representing a range of defect appearances and imaging conditions, it is hypothesized that SegGPT could develop a more comprehensive and nuanced understanding of the target segmentation task. This multi-prompt strategy could potentially mitigate the performance trade-offs associated with single-prompt optimization, enabling the model to achieve consistently high segmentation accuracy across both typical and edge case defect scenarios. Imagine, for instance, providing not just one, but several exemplar defect instances from the same defect category, each capturing subtle variations in appearance due to lighting, imaging artifacts, or defect morphology. Such a richer prompt context could equip SegGPT with the capacity to extract these variations and generalize more effectively to unseen defect instances. Other crucial limitations and future research directions warrant careful consideration. Furthermore, while the qualitative results are encouraging, a rigorous and exhaustive quantitative evaluation is indispensable to objectively benchmark SegGPT's performance against established segmentation methodologies and to provide a statistically positive assessment of its effectiveness in the semiconductor defects domain. Quantitative evaluation should study a diverse range of standard segmentation metrics, such as Intersection over Union (IoU) and Dice score, and should be conducted on a sufficiently large and representative dataset of semiconductor defect images. In conclusion, this project has successfully demonstrated the applicability of the generalist SegGPT model to the specialized and critical task of semiconductor defect segmentation. The proof of concept implementation and the insightful qualitative analysis of its performance characteristics lay a solid foundation for future research and development. The in-context learning capabilities of SegGPT, particularly its potential for adaptation with limited labeled data, align remarkably well with the challenges and constraints prevalent in industrial image analysis, especially within the semiconductor manufacturing sector. While limitations related to prompt sensitivity, computational cost, and the need for comprehensive quantitative validation remain, the findings of this project strongly suggest that SegGPT, hold significant promise for transforming defect detection and segmentation processes in industrial domains, building the way for more efficient, adaptable, and data-conscious approaches to quality control and process optimization in advanced manufacturing. The exploration of multi-prompt learning, coupled with ongoing efforts to enhance computational efficiency and rigorous quantitative evaluation, represents the next steps in realizing the full potential of SegGPT and similar models for real-world industrial deployment and impact. The journey from this promising proof of concept to a robust and deployable industrial solution is complex, but the initial findings presented in this work provide a compelling and optimistic outlook for the future of AI-powered defect detection in semiconductor manufacturing and beyond.

REFERENCES

[1] Philip Bachman, R Devon Hjelm, and William Buchwalter. "Learning representations by maximizing mutual information across views". In: *Advances in neural information processing systems* 32 (2019).

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

[3] Amir Bar et al. “Visual prompting via image inpainting”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 25005–25017.

[4] Leonardo Barcellona et al. “Show and Grasp: Few-shot Semantic Segmentation for Robot Grasping through Zero-shot Foundation Models”. In: *arXiv preprint arXiv:2404.12717* (2024).

[5] Floris van Beers et al. “Deep neural networks with intersection over union loss for binary image segmentation”. In: *Proceedings of the 8th international conference on pattern recognition applications and methods*. SciTePress. 2019, pp. 438–445.

[6] Rishi Bommasani et al. “On the opportunities and risks of foundation models”. In: *arXiv preprint arXiv:2108.07258* (2021).

[7] Tom Brown et al. “Language models are few-shot learners”. In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[8] Attila Budai et al. “Robust vessel segmentation in fundus images”. In: *International journal of biomedical imaging* 2013.1 (2013), p. 154860.

[9] Nico Catalano and Matteo Matteucci. “Few shot semantic segmentation: a review of methodologies and open challenges”. In: *arXiv preprint arXiv:2304.05832* (2023).

[10] Hao Chen et al. “Pixel matching network for cross-domain few-shot segmentation”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2024, pp. 978–987.

[11] Jiaxin Chen et al. “A closer look at the training strategy for modern meta-learning”. In: *Advances in neural information processing systems* 33 (2020), pp. 396–406.

[12] Liang-Chieh Chen et al. “Encoder-decoder with atrous separable convolution for semantic image segmentation”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 801–818.

[13] Xinlei Chen and Kaiming He. “Exploring simple siamese representation learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 15750–15758.

[14] Marius Cordts et al. “The cityscapes dataset for semantic urban scene understanding”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 3213–3223.

[15] Jacob Devlin. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).

[16] Nanqing Dong and Eric P Xing. “Few-shot semantic segmentation with prototype learning.” In: *BMVC*. Vol. 3. 4. 2018, p. 4.

[17] Alexey Dosovitskiy et al. “Discriminative unsupervised feature learning with convolutional neural networks”. In: *Advances in neural information processing systems* 27 (2014).

[18] Mark Everingham et al. “The pascal visual object classes challenge: A retrospective”. In: *International journal of computer vision* 111 (2015), pp. 98–136.

[19] Haoran Fan et al. “DARNet: Bridging Domain Gaps in Cross-Domain Few-Shot Segmentation with Dynamic Adaptation”. In: *arXiv preprint arXiv:2312.04813* (2023).

[20] Qi Fan et al. “Self-support few-shot semantic segmentation”. In: *European Conference on Computer Vision*. Springer. 2022, pp. 701–719.

[21] Muhammad Moazam Fraz et al. “An ensemble classification-based approach applied to retinal blood vessel segmentation”. In: *IEEE Transactions on Biomedical Engineering* 59.9 (2012), pp. 2538–2548.

[22] Tianyi Gao et al. “Enrich Distill and Fuse: Generalized Few-Shot Semantic Segmentation in Remote Sensing Leveraging Foundation Model’s Assistance”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 2771–2780.

[23] Alberto Garcia-Garcia et al. “A survey on deep learning techniques for image and video semantic segmentation”. In: *Applied Soft Computing* 70 (2018), pp. 41–65.

[24] Albert Gu and Tri Dao. “Mamba: Linear-time sequence modeling with selective state spaces”. In: *arXiv preprint arXiv:2312.00752* (2023).

[25] Yanming Guo et al. “A review of semantic segmentation using deep neural networks”. In: *International journal of multimedia information retrieval* 7 (2018), pp. 87–93.

[26] Sina Hajimiri et al. “A strong baseline for generalized few-shot semantic segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 11269–11278.

[27] Bharath Hariharan et al. “Simultaneous detection and segmentation”. In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VII 13*. Springer. 2014, pp. 297–312.

[28] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[29] Sunghwan Hong et al. “Cost aggregation with 4d convolutional swin transformer for few-shot segmentation”. In: *European Conference on Computer Vision*. Springer. 2022, pp. 108–126.

[30] AD Hoover, Valentina Kouznetsova, and Michael Goldbaum. “Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response”. In: *IEEE Transactions on Medical imaging* 19.3 (2000), pp. 203–210.

[31] Gabriel Huang et al. “A survey of self-supervised and few-shot object detection”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.4 (2022), pp. 4071–4089.

[32] Suvarna Kadam and Vinay Vaidya. “Review and analysis of zero, one and few shot learning approaches”. In: *Intelligent Systems Design and Applications: 18th International Conference on Intelligent Systems Design and Applications (ISDA 2018) held in Vellore, India, December 6-8, 2018, Volume 1*. Springer. 2020, pp. 100–112.

[33] Tero Karras. “A Style-Based Generator Architecture for Generative Adversarial Networks”. In: *arXiv preprint arXiv:1812.04948* (2019).

[34] Tero Karras et al. “Analyzing and improving the image quality of stylegan”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 8110–8119.

[35] Diederik P Kingma. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[36] Alexander Kirillov et al. “Segment anything”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 4015–4026.

[37] Biao Li et al. “A survey on semantic segmentation”. In: *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE. 2018, pp. 1233–1240.

[38] Gen Li et al. “Adaptive prototype learning and allocation for few-shot segmentation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 8334–8343.

[39] Xiang Li et al. “Fss-1000: A 1000-class dataset for few-shot segmentation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 2869–2878.

[40] Yiwen Li et al. “Few-shot semantic segmentation with self-supervision from pseudo-classes”. In: *arXiv preprint arXiv:2110.11742* (2021).

[41] Xiaodan Liang et al. “Look into person: Joint body parsing & pose estimation network and a new benchmark”. In: *IEEE transactions on pattern analysis and machine intelligence* 41.4 (2018), pp. 871–885.

[42] Tsung-Yi Lin et al. “Microsoft coco: Common objects in context”. In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer. 2014, pp. 740–755.

[43] Yang Liu et al. “Matcher: Segment anything with one shot using all-purpose feature matching”. In: *arXiv preprint arXiv:2305.13310* (2023).

[44] Yongfei Liu et al. “Part-aware prototype network for few-shot semantic segmentation”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*. Springer. 2020, pp. 142–158.

[45] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[46] Zhihe Lu et al. “Simpler is better: Few-shot semantic segmentation with classifier weight transformer”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 8741–8750.

[47] Timo Lüddecke and Alexander Ecker. “Image segmentation using text and image prompts”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 7086–7096.

[48] Lingchen Meng et al. “SegIC: Unleashing the Emergent Correspondence for In-Context Segmentation”. In: *European Conference on Computer Vision*. Springer. 2025, pp. 203–220.

[49] Juhong Min, Dahyun Kang, and Minsu Cho. “Hypercorrelation squeeze for few-shot segmentation”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 6941–6952.

[50] Khoi Nguyen and Sinisa Todorovic. “Feature weighting and boosting for few-shot segmentation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 622–631.

[51] Atsuro Okazawa. “Interclass prototype relation for few-shot segmentation”. In: *European Conference on Computer Vision*. Springer. 2022, pp. 362–378.

[52] Maxime Oquab et al. “Dinov2: Learning robust visual features without supervision”. In: *arXiv preprint arXiv:2304.07193* (2023).

[53] Alec Radford. “Improving language understanding by generative pre-training”. In: (2018).

[54] Alec Radford et al. “Learning transferable visual models from natural language supervision”. In: *International conference on machine learning*. PMLR. 2021, pp. 8748–8763.

[55] Kate Rakelly et al. “Few-shot segmentation propagation with guided networks”. In: *arXiv preprint arXiv:1806.07373* (2018).

[56] Vignesh Ramanathan et al. “Paco: Parts and attributes of common objects”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 7141–7151.

[57] 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.

[58] Wenqi Ren et al. “Visual semantic segmentation based on few/zero-shot learning: An overview”. In: *IEEE/CAA Journal of Automatica Sinica* (2023).

[59] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. “Stochastic backpropagation and approximate inference in deep generative models”. In: *International conference on machine learning*. PMLR. 2014, pp. 1278–1286.

[60] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer. 2015, pp. 234–241.

[61] Oindrila Saha, Zezhou Cheng, and Subhransu Maji. “GANORCON: Are generative models useful for few-shot segmentation?” In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 9991–10000.

[62] Tomoya Sakai et al. “A Surprisingly Simple Approach to Generalized Few-Shot Semantic Segmentation”. In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024.

[63] Amirreza Shaban et al. “One-shot learning for semantic segmentation”. In: *arXiv preprint arXiv:1709.03410* (2017).

[64] Xinyu Shi et al. “Dense cross-query-and-support attention weighted mask aggregation for few-shot segmentation”. In: *European Conference on Computer Vision*. Springer. 2022, pp. 151–168.

[65] Mennatullah Siam and Boris Oreshkin. “Adaptive masked weight imprinting for few-shot segmentation”. In: (2019).

[66] Karen Simonyan. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[67] Jake Snell, Kevin Swersky, and Richard Zemel. “Prototypical networks for few-shot learning”. In: *Advances in neural information processing systems* 30 (2017).

[68] Haoliang Sun et al. “Attentional prototype inference for few-shot segmentation”. In: *Pattern Recognition* 142 (2023), p. 109726.

[69] Pinzhuo Tian et al. “Differentiable meta-learning model for few-shot semantic segmentation”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 07. 2020, pp. 12087–12094.

[70] Zhuotao Tian et al. “Prior guided feature enrichment network for few-shot segmentation”. In: *IEEE transactions on pattern analysis and machine intelligence* 44.2 (2020), pp. 1050–1065.

[71] Nontawat Tritrong, Pitchaporn Rewatbowornwong, and Supasorn Suwajanakorn. “Repurposing gans for one-shot semantic part segmentation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 4475–4485.

[72] Oriol Vinyals et al. “Matching networks for one shot learning”. In: *Advances in neural information processing systems* 29 (2016).

[73] Haochen Wang et al. “Few-shot semantic segmentation with democratic attention networks”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*. Springer. 2020, pp. 730–746.

[74] Haochen Wang et al. “Variational prototype inference for few-shot semantic segmentation”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2021, pp. 525–534.

[75] Jin Wang et al. “Rethinking Prior Information Generation with CLIP for Few-Shot Segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 3941–3951.

[76] Junjue Wang et al. “LoveDA: A remote sensing land-cover dataset for domain adaptive semantic segmentation”. In: *arXiv preprint arXiv:2110.08733* (2021).

[77] Kaixin Wang et al. “Panet: Few-shot image semantic segmentation with prototype alignment”. In: *proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 9197–9206.

[78] Xinlong Wang et al. “Images speak in images: A generalist painter for in-context visual learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 6830–6839.

[79] Xinlong Wang et al. “Seggpt: Towards segmenting everything in context”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 1130–1140.

[80] Yaqing Wang et al. “Generalizing from a few examples: A survey on few-shot learning”. In: *ACM computing surveys (csur)* 53.3 (2020), pp. 1–34.

[81] Zhaoqing Wang et al. “Cris: Clip-driven referring image segmentation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 11686–11695.

[82] Syed Waqas Zamir et al. “isaid: A large-scale dataset for instance segmentation in aerial images”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2019, pp. 28–37.

[83] Zhonghua Wu et al. “Learning meta-class memory for few-shot semantic segmentation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 517–526.

[84] Pengfei Xian et al. “Clip driven few-shot panoptic segmentation”. In: *IEEE Access* 11 (2023), pp. 72295–72305.

[85] Boyu Yang et al. “Prototype mixture models for few-shot semantic segmentation”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VIII 16*. Springer. 2020, pp. 763–778.

[86] Chi Zhang et al. “Canet: Class-agnostic segmentation networks with iterative refinement and attentive few-shot learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 5217–5226.

[87] Chi Zhang et al. “Pyramid graph networks with connection attentions for region-based one-shot semantic segmentation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 9587–9595.

[88] Renrui Zhang et al. “Personalize segment anything model with one shot”. In: *arXiv preprint arXiv:2305.03048* (2023).

[89] Xiaolin Zhang et al. “Rich embedding features for one-shot semantic segmentation”. In: *IEEE Transactions on Neural Networks and Learning Systems* 33.11 (2021), pp. 6484–6493.

[90] Xiaolin Zhang et al. “Sg-one: Similarity guidance network for one-shot semantic segmentation”. In: *IEEE transactions on cybernetics* 50.9 (2020), pp. 3855–3865.

[91] Yuxuan Zhang et al. “Datasetgan: Efficient labeled data factory with minimal human effort”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 10145–10155.

[92] Hengshuang Zhao et al. “Pyramid scene parsing network”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2881–2890.

[93] Ziqin Zhou et al. “Zegclip: Towards adapting clip for zero-shot semantic segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 11175–11185.

[94] Lianghui Zhu et al. “Vision mamba: Efficient visual representation learning with bidirectional state space model”. In: *arXiv preprint arXiv:2401.09417* (2024).