# On the Generalizability of Deep Learning-based Medical Image Segmentation Methods

Birk Sebastian Frostelid Torpmann-Hagen


Thesis submitted for the degree of
Master in Robotics and Intelligent Systems
60 credits

Department of Informatics
Faculty of mathematics and natural sciences

UNIVERSITY OF OSLO

Spring 2022

# On the Generalizability of Deep Learning-based Medical Image Segmentation Methods

Birk Sebastian Frostelid Torpmann-Hagen

On the Generalizability of Deep Learning-based Medical Image Segmentation Methods

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

# Abstract

Despite achieving state-of-the-art performance in lab-conditions, deep learning-based systems often exhibit significant performance degradation when deployed in practical settings. This is referred to as *generalization failure*. Why and how this occurs has only recently started to be understood, and there has consequently been limited research towards developing generalizable methods for deep learning.

This thesis attempts to address generalization failure in the domain of medical image segmentation, in particular on the polyp segmentation task. Recent analyses of generalizability is discussed, which is then used to inform the development of a number of novel methods. This includes a simple dual-decoder architecture, an augmentation strategy which incorporates a generative polyp inpainter, a training method referred to as *Consistency Training*, and finally, several ensemble models for which the constituent predictors are trained using Consistency Training.

These methods are then evaluated through multiple quantitative studies. As the extent to which methods used as baselines in this thesis affect generalization is not particularly well understood, this thesis also contributes a quantitative analysis of the the impact of the choice of model architecture, data augmentation, and ensemble-models on generalization.

The results show that Consistency Training facilitates increased generalization over data augmentation. The use of the inpainter as a component of data augmentation, however, limits the possible improvements compared to regular augmentation. Ensembles improve generalization, albeit to a somewhat lesser extent than the aforementioned methods. Finally, the choice of model architecture, including the use of a secondary decoder, is shown to have negligible effects on generalization. These results were all explored with respect to theory presented in other literature.

These findings are then analyzed and used to inform a number of hypotheses which are suggested as points of further study. Several improvements to the proposed methods were also suggested, in particular with regards to Consistency Training, which shows significant promise towards further mitigating generalization failure.

# Contents

# List of Figures

# List of Tables

# Acknowledgments

I would like to thank my supervisors Michael Riegler, Vajira Thambawita, Pål Halvorsen and Kyrre Glette for their insightful feedback and for providing me with such an interesting topic of research. I would also like to thank my fellow students in room 4104 for entertaining my many technical discussions. Finally, I would like to thank my family for their support and interest in my work.

The research presented in this paper has benefited from the Experimental Infrastructure for Exploration of Exascale Computing (eX3), which is financially supported by the Research Council of Norway under contract 270053.

# Chapter 1

# Introduction

The last decade or so has seen a veritable revolution in Artificial Intelligence (AI). This has been spearheaded principally by advancements in Deep Learning, the remarkable performance of which has rendered more conventional approaches practically obsolete[91]. Recent work has, however, highlighted that models trained using deep learning, i.e., Deep Neural Networks (DNNs), are highly prone to exhibiting significant reductions in performance when deployed in practical settings despite readily exhibiting high performance when evaluated on previously unseen subsets of the training data [20, 29, 37, 42]. This is referred to as *generalization failure*.

This type of behaviour is especially prevalent in the domain of medical imaging. Though medical imaging in recent years has proven to be one of the most promising applications of deep learning, having the capacity to significantly improve both the accuracy and efficiency of the detection, diagnosis, and treatment of a wide variety of diseases [84], recent research has shown that these types of systems are particularly susceptible to generalization failure. Whereas other domains often have access to exceedingly large datasets, medical datasets are typically fairly small both due to privacy concerns and the high costs associated with annotation. Moreover, even when large datasets are available, they are unlikely to fully encapsulate the nature of whatever relationships they are intended to represent due to the inherent variability present in medical domains. There are often high degrees of variability within the same class of pathology, which in addition to multitudes of confounding variables, such as differences in clinical routines, demographics, imaging equipment, and so on typically result in DNNs exhibiting high degrees of sensitivity to even minor changes in the nature of the input data. Finally, since medical datasets are typically collected from a single hospital, from a limited demographic, and with a limited selection of equipment, sampling bias is practically unavoidable [52, 58]. In addition to the fact that this reduces the overall clinical utility of the system, it may also induce certain societal consequences if deployed [69].

This all exacerbates the already difficult task of making deep learning systems sufficiently generalizable for practical use. Indeed, even DNNs trained on enormous non-medical datasets exhibit high degrees of sensitivity to distributional shifts, for example due to changes in texture [28], additive noise [42], and minor image corruptions [37]. Even perturbations imperceptible to the human eye in the form of *adversarial attacks* can break even the most sophisticated deep learning systems [13].

Thus, ensuring that the performance of deep-learning-based systems is generalizable and sufficiently robust is a matter of particular importance in medical domains. Whereas more general deep learning pipelines can sometimes avoid generalization failure by virtue of the sheer volume of training data available, the constraints imposed by the medical domain necessitate more carefully designed pipelines, with a particular focus on ensuring maximal generalizability.

Conventional implementations of deep-learning systems tend to neglect this fact, with high performance on unseen subsets of the training dataset typically being considered a sufficient indicator of generalization. This is highly misleading as to the actual performance of the given model should it be deployed in a setting where the nature of the data may differ, even if the differences between the two domains are slight or even unremarkable to a human observer. Such data is often referred to as being *Out of Distribution (OOD)*. There has been limited research addressing the development of methods that facilitate generalization to such data, in large part because the mechanisms behind generalization failure are currently only in the beginning stages of being understood to an actionable degree.

The EndoCV2021 competition [3] was organized in order to motivate such research in the context of detection- and segmentation of colorectal polyps. This body of work and the advancements it brought, along with the multitude of different datasets available in this domain, means that polyp segmentation constitutes a compelling candidate for a case-study towards understanding generalization failure and developing generalizable methods. The polyp segmentation task will as a consequence serve as the primary context of the work presented in thesis.

## 1.1 Case study: Colon Polyp Segmentation

Colorectal cancer is one of the leading causes of cancer related deaths, causing approximately 900 000 deaths worldwide per year [24]. Early detection and subsequent resection of polyps, a precursor to colorectal cancer, is therefore of significant importance towards reducing the incidence- and mortality-rates thereof. Polyps are, however, easily missed during colonoscopies due to the significant variability in their shapes and sizes, as well as the high degree of visual similarity to surrounding tissue [38, 74].

Automatic segmentation of polyps via deep learning has as a consequence been identified as a promising candidate for reducing polyp miss-rates by serving as an auxiliary detection method during screening. There has been a wealth of work dedicated to developing such systems [2, 3, 39, 89], with some studies reporting that AI-assisted detection may increase detection rates by 10% in clinical deployment [9].

As mentioned, however, these sorts of systems have also been shown to be highly prone to generalization failure [1, 20, 29]. To address this, the EndoCV2021 competition [3] was organized with the primary goal of developing generalizable deep learning systems for polyp-segmentation and -detection. The submissions were evaluated on several unseen datasets consisting of endoscopic images collected from a separate center than the provided training datasets as well as images collected using a differing endoscopic lighting system. Though several teams made good progress towards increasing generalizability, the organizers' review of the submissions [1] highlighted that every submitted model nevertheless exhibited significant performance reductions on the aforementioned unseen datasets.

Furthermore, there has at the time of writing this thesis been limited research dedicated to developing an understanding of the relative impact of the many design elements present in most deep learning pipelines on generalization. Indeed, most analyses performed today, therein those performed in the EndoCV2021 review[1], do not explicitly control potentially highly affecting variables - such as the choice of data augmentation strategies - when comparing methods, instead only considering the performance of the final system. As a consequence, there is a somewhat limited understanding of the relative impacts of the many methods and techniques believed to improve generalization.

## 1.2 Research Objectives

This thesis aims to build upon and synthesize the findings reported in EndoCV2021 and other recent works on generalizability. The overall goal of is as such to explore methods of increasing the generalizability of deep-learning-based polyp-segmentation systems. This goal can be decoupled into the following pair of research objectives:

1. **To leverage recent advances in the understanding of generalization failure to inform the development of novel methods of increasing the generalization of deep learning systems for polyp-segmentation**. By synthesizing the often fragmented analyses of generalization failure presented in the literature, one can develop a more holistic understanding of why these failures occur and the mechanisms behind them. This facilitates the development of more targeted methods towards increasing generalizability.

2. **To synthesize recent work on generalizability and determine concretely the degree to which conventional and well-established methods affect generalization**. Deep Learning systems are highly complex, with several moving parts and complicated dynamics. Analyzing the impact of the constituent components thereof on generalization is therefore warranted. In particular, this thesis compares the impact of the following variables: the choice of model architecture, the use of data augmentation, and the use of ensembles, as these methods can broadly be considered the most common subjects of research on generalizable methods. Among the eight submissions to the segmentation portion of EndoCV2021, for instance, three primarily made use of ensembles [41, 57, 88], two developed novel model-architectures [27, 36], and one developed an augmentation strategy [30].

## 1.3 Main Contributions and Key Findings

Objective 1 was achieved through the development of the following novel methods:

- A framework for analyzing generalizability based on reframing it as the model's ability to output predictions that are consistent across distributional shifts.
- A metric and loss function intended to quantify this notion of consistency in the context of segmentation, referred to as Segmentation Inconsistency Score (SIS) and Segmentation Inconsistency Loss (SIL) respectively.
- A custom augmentation strategy intended to induce the aforementioned distributional shifts in a controlled manner, leveraging both conventional augmentations and a Generative Adversarial Network (GAN) which generates synthetic polyps in a given image.
- A training paradigm which makes use of the aforementioned framework, loss function and augmentation strategy, referred to as Consistency Training. In contrast to many competing methods, this does not require multiple datasets, and is for practical purposes a more generalizable alternative to data augmentation. This method was also the basis for a research paper submitted to NeurIPS 2022, which can be found in Appendix D.
- Several ensembles consisting of models trained according to the aforementioned training paradigm.
- A simple dual-decoder model, wherein one decoder performs image reconstruction and the other segmentation. This is intended to facilitate the learning of more generalizable features.

These methods were then evaluated through four separate experiments. To fully understand the relative impacts of these methods, several baselines were also implemented, varying the model architecture, augmentation strategy, and the use of ensembles. These baselines were then compared both to the novel methods as presented above, and to one another in order to ascertain the individual impacts, hence achieving Objective 2.

The results from these experiments were then analyzed with respect to theoretical frameworks presented in Chapter 2. The findings from these analyses then informed a number of hypotheses which were suggested as points of further study. The most notable findings can be summarized as follows:

- Consistency training greatly increased generalizability, outperforming every other tested method. Several possible improvements to Consistency Training were also presented, along with ideas for more advanced training methods that make use of the Consistency framework.
- Data augmentation also increased generalizability, albeit by a somewhat smaller margin than Consistency Training. When the augmentation strategy incorporated a generative inpainter, the gains were marginally less substantial. It was argued that the extent to which the use of data augmentation affects generalization raises questions as to the veracity of comparative studies that do not account for the use of disparate augmentation strategies, therein EndoCV2021.
- Ensemble models improve generalization by a minor amount when compared to the mean performance of the models that make them up. Similar improvements could be observed regardless of the training procedure and model architecture used, suggesting perhaps unsurprisingly that the generalizability of ensembles is primarily determined by the generalizability of the constituent models. It was also shown that the gains from ensembles is correlated with the variability in performance between the constituent models. A diversity-based training method for ensemble models was suggested to investigate this further.
- The model architectures tested in this thesis all exhibited fairly comparable degrees of generalizability. The introduced dual-decoder model did not contribute to increased generalization. After analyzing this result, it was theorized that segmentation encoders learn task-invariant features and thus can be interpreted as primarily performing image compression. Further experiments were suggested to investigate this.

Though this work by no means solves the problem of generalization failure, the aforementioned contributions constitute a significant step in the right

direction. For one, the results and analyses performed in this thesis provide a holistic perspective of the impact of the tested baseline methods on generalization. Secondly, the novel methods presented were shown to have significant potential for continued research towards further understanding generalization failure and increasing generalizability. Consistency Training in particular was proven to be a highly promising concept, with plenty of room for further development.

## 1.4 Research Methods

The research methods used in this thesis were principally of an exploratory and quantitative nature [56]. The methods were analyzed quantitatively, and the relative performance of each method determined to statistical significance. The analysis of the findings from these comparisons and the resulting theories explaining them with respect to the theory was exploratory, as was the development process for the novel methods.

This approach was chosen due to the inherently dualistic nature of the thesis as per the research objectives; a quantitative approach permits statistically significant comparisons between methods, whereas an exploratory approach affords flexibility with regards to the development of the methods as well as permitting sufficient analysis of the quantitative findings with respect to the established theoretical frameworks.

## 1.5 Organization of the Thesis

The thesis will be organized as follows:

- Chapter 2 will cover all relevant background knowledge. This includes a brief introduction to polyps and their role in colorectal cancer, deep learning, and segmentation, as well as an overview and synthesis of related works on generalization failure and generalizable methods for deep learning. Ethical considerations pertaining to generalizability will also be discussed.

- Chapter 3 will cover the novel methods that constitute the contributions as outlined above, their basis with respect to the theory presented in Chapter 2, as well as details surrounding their implementation where applicable.

- Chapter 4 will describe the experimental setup, as well as present the experiments and the results thereof. These results will also be explored the context of the theory established in Chapter 2.

- Chapter 5 will discuss these findings, their impacts and their limitations.
- Finally, Chapter 6 summarizes the work done in this thesis along with presenting directions for further research.

# Chapter 2

# Background

Polyps are small growths found in and around the inner lining of the large intestine. These polyps, also referred to as adenomas, can in time develop into cancerous tumors, or carcinomas, in a process known as the adenoma-carcinoma sequence [61]. Though the majority of polyps do not undergo this process, identifying polyps nonetheless constitutes an important step towards preventing colorectal cancer. Indeed, resection of these polyps has been shown to reduce the incidence of colorectal cancer by a significant margin [95].

Though colorectal cancer remains as one of the leading causes of cancer-related death worldwide [24], mortality rates have in recent years declined in large part to the increased use of screening colonoscopy and subsequent preemptive treatment [47]. Polyps are by nature somewhat difficult to detect, however, and are routinely missed by clinicians, with miss rates reportedly ranging upwards of 27% for diminutive (<2.5mm) polyps [38, 74]. Reducing this miss rate has the potential to further reduce the incidence of colorectal cancer. As a result, there has been a significant body of work dedicated to developing systems and techniques to aid in more accurate screening. Certain image-processing techniques, namely I-SCAN, have for instance been shown to reduce miss-rates by up to 50% [14]. Similarly, the use of narrow-band imaging, wherein light of specific wavelengths specifically designed to highlight the textural differences between the polyps and the surrounding tissue, have been shown to reduce miss rates by 26% [15].

These systems do, however, require specialized equipment, training and expertise to effectively employ. Thus, automatic polyp detection using DNNs, and in particular Convolutional Neural Networks (CNNs), has also been identified as a possible ancilliary screening method. This requires minimal training time on the part of the clinician, no additional equipment, and has been shown to increase detection rates by 10% when deployed in a clinical setting [9].

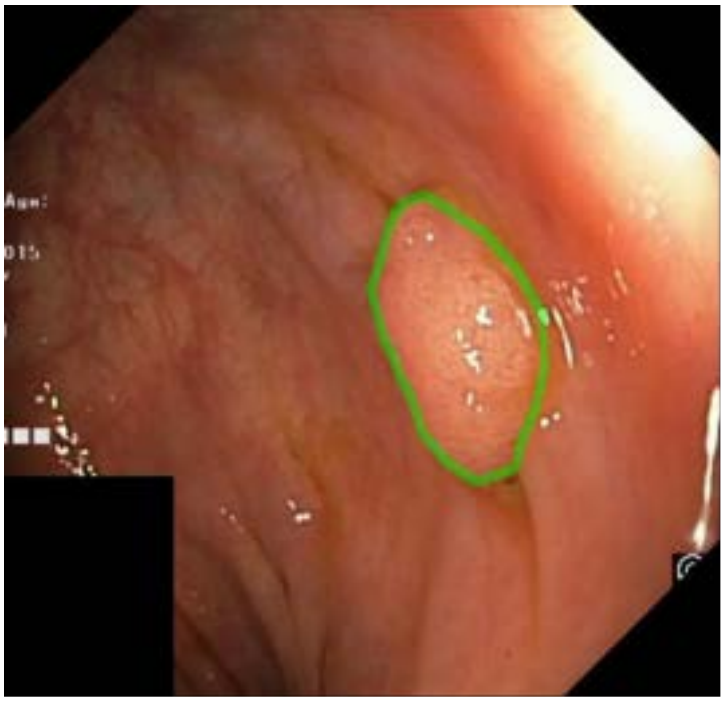

Figure 2.1: Example of a colorectal polyp from the Kvasir-SEG [49] dataset. The polyp is outlined in green.

This has spurred on a large body of research dedicated to improving the performance and expanding the capabilities of deep-learning based systems for polyp detection and segmentation. Several challenges have also been held, namely the Endotect Challenge [39], EndoCV2020 [2] and EndoCV2021 [3].

There are, however, still several hurdles to overcome. Recent research has shown that even state of the art deep-learning pipelines are prone to generalization failure when deployed in practical settings, particularly when exposed to distributional shifts such as changes in demographics, imaging equipment, noise, and more, despite exhibiting high performance on hold-out sets [10, 20, 29, 99]. This was further highlighted in the EndoCV2021 challenge, wherein submissions were evaluated on OOD datasets collected from different centers or imaging methods than the training data. Though several teams made good progress towards increasing generalizability, the organizers' review of the submissions [1] highlighted that every submitted model nevertheless exhibited significant performance reductions on the provided OOD datasets, demonstrating the difficulty involved in developing generalizable models.

Understanding how and why such generalization failure occurs and developing methods to counteract it is a subject of ongoing study. This chapter will summarize and synthesize recent findings in the field. It will first cover the necessary understanding of deep learning and segmentation, be-

fore moving on to a survey of instances of generalization failure both in systems dedicated to polyp-segmentation and other applications of deep learning. These failures will then be analyzed through the lens of generalizability theory, starting from the theoretical fundamentals underpinning deep learning - namely Empirical Risk Minimization (ERM) - and incorporating recent analyses in the literature pertaining to generalizability failure and its origins. Finally, recent work on generalizable methods will be summarized, and analyzed with respect to the aforementioned theory.

## 2.1 Deep Learning

The past decade or so has seen considerable advancements in Deep Learning. This has facilitated significant performance improvements for a variety of different tasks and domains, including computer vision [91], finance [43], natural language processing and machine translation [70], content recommendation engines [22], robot-control [72], and games like Chess and Go [87].

To fully explain how and why Deep Learning performs so well - and why it sometimes does not - this section will cover the basics of Deep Learning and the Deep Learning Pipeline. It will also detail the problem of semantic segmentation in the context of polyps, and finally describe how a Deep Learning Pipeline can be adapted to try to solve this problem.

### 2.1.1 The Deep Learning Pipeline

Deep Learning is a supervised machine learning method, wherein a Deep Neural Network (DNN) - typically consisting of millions and even hundreds of billions of parameters - learns to identify patterns conducive to approximating the mapping between pairs of inputs and labels given by a dataset [32]. Conceptually, one can consider DNNs as general-purpose correlation machines; i.e. they accept some paired input-output data, learn correlations between the inputs and outputs and then predict according to these correlations at inference-time. Similar to how one can establish linear relationships via linear regression on a set of (linearly) related variables, DNNs are capable of establishing non-linear relationships via Deep Learning on arbitrarily related sets of paired input-output data. The inputs can for instance be images and the outputs categories (also referred to as classes), bounding boxes, or -as in the case of segmentation - image regions, or the inputs could be sentences in English and the outputs sentences in French. So long as the data can be encoded into a vector-space, deep learning can typically be applied.

This is achieved through a process known as training, the objective of

which is to adjust the parameters of the DNN such that the model exhibits maximal performance. Training a DNN is not straight-forward; each parameter corresponds to a dimension in the search space, and searching through millions upon millions or more dimensions in order to find a sufficiently performant parameter configuration is a challenging problem. Deep Learning systems nevertheless achieve this through a process known as gradient descent [77]. Fundamentally, this involves minimizing some quantity inversely proportional to whatever performance metric one seeks to maximize. This quantity is referred to as the *loss*, and the function that generates it a *loss function*. Minimizing the loss is achieved by differentiating the loss function with respect to the model's parameters and adjusting them in the direction of the gradient. There are a number of complicating factors involved in this process, which through nearly a decade of research have been addressed using a number of different techniques culminating in what will be referred to as the *deep learning pipeline*. The constituent components thereof, as well as further technical details as to how DNNs are trained, will be further described in the following paragraphs, and are illustrated in Figure 2.2.

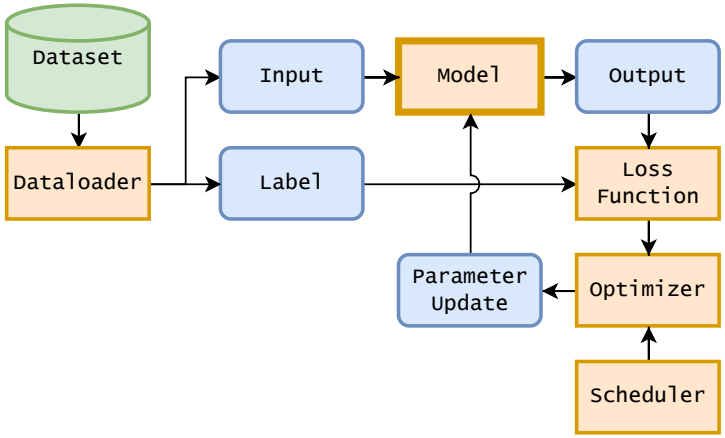


Figure 2.2: A Conventional Deep Learning pipeline.

### 2.1.2 Architectures and Models

The DNN, typically referred to simply as the *model*, can be considered the central component of the deep learning pipeline. There are several types of DNNs, including Recurrent Neural Networks (RNNs) [64], Transformers [90], Multi-Layer Perceptrons (MLPs) [63], and CNNs [4]. Common to all of the aforementioned types is that they consist of multiple instances of similar functional blocks, often called *layers*, which are connected to one another. A MLP consists of layers of perceptrons, a RNN primarily consists of stacked recurrent units, a transformer primarily consists of multiple scaled dot-product attention blocks, and a CNN consists primarily of convolutional layers. In computer vision tasks, therein

polyp-segmentation, primarily CNNs are used, though other architectural components can and often are used in conjunction therewith [91].

Convolutional layers are, as their name suggests, based on the convolution operator. Convolution lends itself well to image-related tasks, as it exhibits translational invariance, and is endowed with the ability to consider context by virtue of the fact that convolutions operate with sliding windows. This is illustrated Figure 2.3.

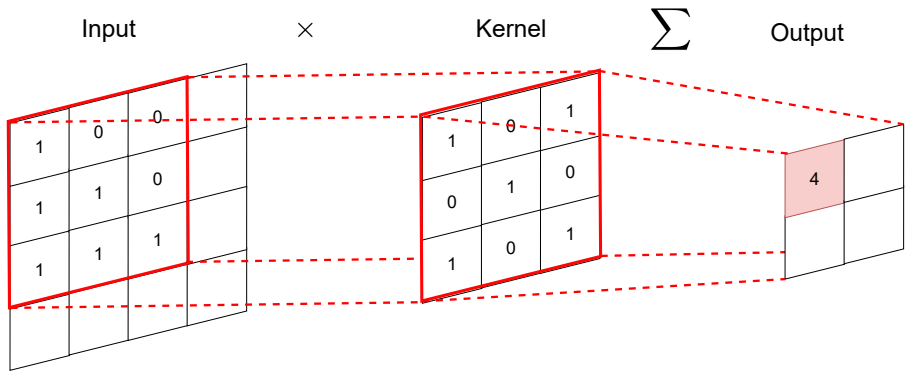


Figure 2.3: Example of Convolution

How large of a context that a given network (or layer) considers for a given pixel is referred to as its receptive field. By stacking convolutional layers, one can multiplicatively increase the receptive field of the network. This, in conjunction with injecting non-linearities through so-called activation functions between each convolutional layer, allows CNNs to learn both highly non-linear and highly context-dependent relationships from the data.

Optimally, each layer in fully trained network will encode increasingly complex representations of the data. This set of representations is called the latent space of the network. The properties that the representations encode are referred to as the model's learned features. In theory, the deeper the network, the larger the latent space, and therefore the more complex features can be encoded. This enables deep convolutional networks to significantly outperform computer vision systems developed using more conventional approaches, such as the usage of feature engineering methods in conjunction with Random Forests, Support Vector Machines or other simpler classifiers. Instead of having to manually engineer features for a given task, CNNs simply learn to generate optimal features automatically.

### 2.1.3 Training and Gradient Descent

In order for a model to do anything useful, it first has to learn the relationship between the inputs and the labels in the dataset it is given. This

is referred to as training the model. Conventional Deep Learning pipelines achieve this through an algorithm known as Empirical Risk Minimization (ERM) by gradient descent. The details behind this process and more precise formulations will be covered in later sections and be related to generalization, but for now a high-level view is sufficient.

Gradient descent is an optimization procedure whereby one seeks to minimize a *loss-function* $\mathcal{L}(\cdot, \cdot)$, a differentiable distance function which quantifies how wrong the model is when compared to label data. This is achieved as follows: first some number of input-label pairs $x_i, y_i$ are selected from the dataset, via a *dataloader*. The dataloader determines how the inputs will be processed - i.e if they require scaling, shuffling, or augmentation - as well as the number of samples that the remaining pipeline will incorporate into gradient calculations - referred to as the *batch size*. The inputs are then passed through the *model*, often denoted simply as $f(\cdot)$, generating outputs $f(x_i) = \hat{y}_i$. Afterwards, the loss is computed by comparing the output and labels according to the loss function evaluated at the current outputs $\mathcal{L}(y, \hat{y})$. This is then differentiated with respect to each of the model's parameters $W$ in a process known as back-propagation. This yields a series of vectors for each set of the weights, which correspond to the direction in the parameter space that would result in the most increase to the loss function. This is called the gradient, and is denoted as $\nabla_W \mathcal{L}(\cdot)$. Equivalently, the negative gradient corresponds to the direction which would result in the biggest reduction of the loss function.

The gradient is, however, only a direction, and does not on its own hold any information regarding by how much the weights should be updated, only the direction in the search space that the update should be sampled from. The magnitude of the update vector is instead decided by two components in the pipeline: the *optimizer* and the *scheduler*. The optimizer dynamically determines the magnitude of the weight update - i.e by how much the gradient is scaled - for instance by considering running averages of recent histories of gradient magnitudes [54]. This magnitude is then scaled once more according to a *learning-rate* $\eta$. The scheduler, in turn, modulates this learning rate according to some predetermined function. This is then repeated for all the data in a dataset.

Iterating over the dataset once is not typically sufficient to arrive at a parameter configuration with desirable performance, however. As a result, this process is typically repeated a set number of times, often referred to as the number of *epochs*.

There are some caveats to this, in particular regarding the generalizability of the resulting model. In particular, each component of the pipeline can be implemented with some form of *regularization*, which in simple terms serves to affect the training procedure such that local minima are avoided by injecting noise. Regularization can take many forms, and can be implemented in of practically any component of the pipeline:

The dataloader can perform data augmentation, the loss function may incorporate regularizing terms such as L2 penalties [19], the model may incorporate dropout connections [40] or batch normalization[46], the optimizer may have weight decay terms [55], and so on. These factors and their effect on training will be discussed further in Section 2.4.

## 2.2 Segmentation

Segmentation is the task of determining the region(s) in image-space that correspond to some relevant classification target. For polyp-segmentation, for instance, this involves marking whatever pixels correspond to polyps. An example is shown in Figure 2.4.

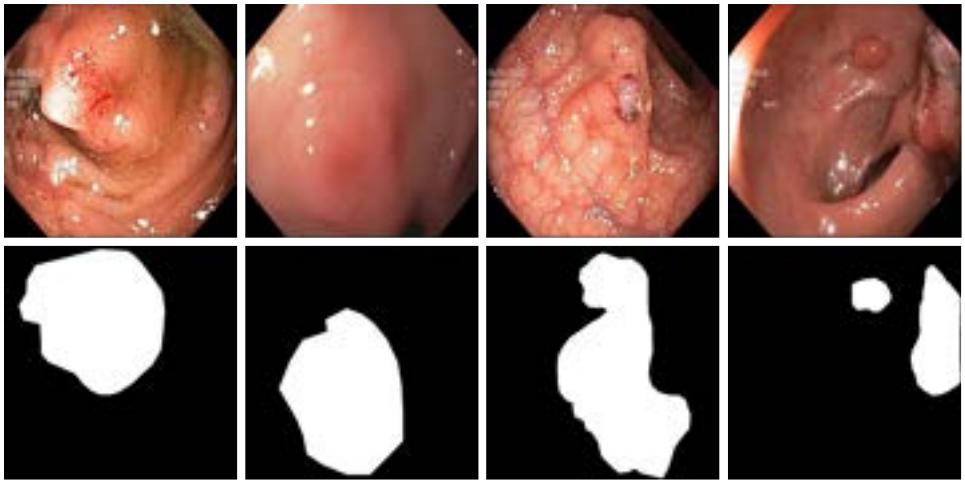

Figure 2.4: Polyp Segmentation Examples, taken from Kvasir-SEG[49]

There are two types of segmentation: semantic segmentation and instance segmentation [66]. In instance segmentation, every instance of the objects require their own segmentation mask and label. In semantic segmentation, only the class of the relevant objects are considered, and multiple instances are considered in unison. I.e, if the task is to segment people in a crowd, instance segmentation will attempt to generate multiple masks, one for each individual, whereas a semantic segmentation model would simply generate a single mask for the crowd in its entirety. Though they are similar, these tasks require somewhat different pipelines. Since there is less of a need to distinguish between polyps than simply detecting the presence thereof, polyp segmentation pipelines are typically oriented around semantic segmentation.

This section will cover the specifics required to design a deep learning pipeline for semantic segmentation, including how the models are designed and the loss functions that are typically used.

### 2.2.1 Semantic Segmentation Models

Deep Learning models for semantic segmentation take an image as input, and outputs a set of segmentation masks consisting of probability value(s) that each pixel belongs to the given class(es). There are a wealth of models that have been developed for this purpose, spanning over a wide range of different architectural frameworks, building blocks and processing methods [59, 66]. Though the details regarding how each and every one of these models work is beyond the scope of this thesis, most of these models share certain architectural traits that warrant explanation.

In particular, most segmentation models consider the scene at multiple scales. This is illustrated in Figure 2.5. In encoder-decoder models, for instance, the image is first processed by the encoder, which consists of layers that successively downsample the image through pooling, strided convolutions, or other mechanisms. This yields a highly compressed latent representation of the scene, which (ideally) should contain all the necessary information in the image that is conducive to segmenting the relevant object(s). The decoder then takes this latent representation, and through layers such as deconvolutions, atrous convolutions, pure upsampling, or similar methods generate some number of segmentation masks, one for each class. In the case of polyp segmentation, this would simply be one image, consisting of probabilities that each pixel belongs to the polyp class. If the probability is low, it is likely that the pixel is not a part of a polyp, whereas if the probability is high, it is likely that the pixel is a part of a polyp.

Unets [76] take this encoder-decoder architecture a step further, by concatenating the representations at corresponding depths in the encoder and decoder.

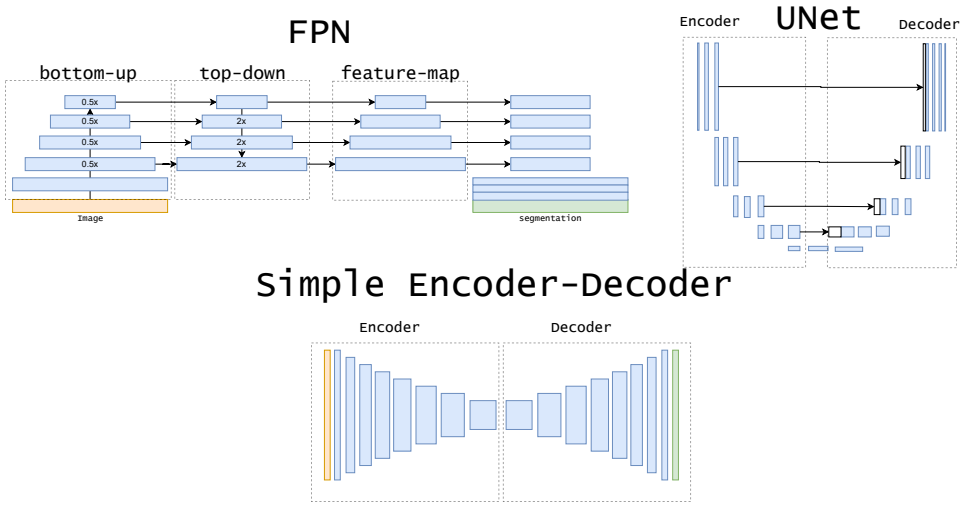


Figure 2.5: Examples of segmentation architectures.

Feature Pyramid Networks (FPNs) work in a similar fashion, but instead consider the input images at multiple scales concurrently, which are then merged at the end of the network such that all scales are considered holistically.

### 2.2.2 Metrics

Segmentation pipelines are typically evaluated by considering the the Dice Coefficient, defined in Equation (2.2), or equivalently Intersection over Union (IoU), defined in Equation (2.1), between the labels and segmentation output, often along with the precision and recall [66].

Accuracy is on its own not very informative, since high accuracies can be achieved simply by predicting all negative if the relevant objects occupy a small portion of the image.

$$IoU(y, \hat{y}) = \frac{\sum\{y = 1\} \cap \{\hat{y} = 1\}}{\sum\{y = 1\} \cup \{\hat{y} = 1\}} \tag{2.1}$$

$$Dice(y, \hat{y}) = \frac{2\sum\{y = 1\} \cap \{\hat{y} = 1\}}{\sum\{y = 1\} + \sum\{\hat{y} = 1\}\}} \tag{2.2}$$

Both the Dice cofficient and the IoU, which sometimes also is referred to as the Jaccard index, can to some extent be interpreted as the accuracy of the segmentation but considered only from the perspective of the regions defined by the respective segmentations, and has the advantage of facilitating easier comparison of models compared to accuracy, which as mentioned tends to be skewed towards high values due to the large proportion of negative pixels in any given image.

Precison and recall, defined in Equation (2.3) and Equation (2.4), respectively, describe the purity and the completeness of the positive predictions. In a polyp-segmentation setting, precision describes the proportion of pixels in a segmented region that do, in fact, constitute a polyp, and recall in effect corresponds to the detection rate on a per-pixel basis.

$$Precison = \frac{TP}{TP + FP} \tag{2.3}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.4}$$

### 2.2.3 Losses

Though there are a number of different loss functions that can be used [59, 65], it is sufficient to consider the four general types they can be categorized as:

- Distributional losses, such as cross-entropy loss. These loss functions quantify statistical properties of the label-output pairs, for instance by calculating their cross-entropy.
- Region-based losses, such as Dice-loss and Jaccard-loss, which instead consider the regions defined by the segmentation labels and outputs. These are, in effect, non-thresholded and thus differentiable versions of the Dice coefficient and IoU.
- Boundary-based losses, such as boundary-loss and HD loss, which consider the boundaries of the segmentation regions.
- Hybrid losses which combine the aforementioned concepts, such as DiceCE, which as its name suggests combines Dice loss and cross-entropy loss.

Asides from the use of these loss functions, model architectures, and evaluation metrics, training is otherwise fairly conventional for a deep learning pipeline. The dataloader provides an image to the model, for which the gradient is computed by descending the gradients of the loss function, with the parameters being adjusted according to the update rule defined by this gradient, the scheduler and the optimizer.

## 2.3 Generalization Failure in the Wild

Recent analyses have showed that DNNs fail to maintain sufficient performance in deployment, even if they exhibit exceptional performance on previously unseen subsets of the training data (holdout sets) [3, 20, 29]. This phenomenon, which has been proven to be commonplace in many applications of deep learning, is known as *generalization failure*. This section will cover some examples of generalization failure across different domains in order to demonstrate the pervasiveness of the problem.

### 2.3.1 Generalization Failure in Medical Imaging

As mentioned in Chapter 1, medical deep learning pipelines are particularly prone to generalization failure due to limited dataset sizes and the sheer difficulty of the tasks involved.

For example, a deep-learning based classifier which successfully detected pneumonia in X-ray scans across a number of hospitals with striking accuracy was determined to be basing its predictions not on any lesions or otherwise pathologically relevant features in the images, but rather on a hospital-specific metal token that could be found in every image, which it used in conjunction with learning the pneumonia prevalence rate

of for the respective hospitals to make predictions. As a result, when deployed on data from hospitals that it had not seen during training, the system failed to generalize [99]. In another study, it was shown that a classifier intended to detect diabetic retinopathy exhibited signiicant performance drops when the model was tested on images taken with a different type of camera [20]. The same study also showed that the performance of a model trained to identify skin-conditions was highly dependent on the skin tone of the subject. Finally, a model trained to detect and diagnose melanomas was shown to in large part be basing its predictions on whether it could detect any pre-surgical markings in the vicinity of the lesion as opposed to actually learning anything about what the melanomas themselves [96]. As these kinds of markings naturally are highly correlated with melanomas, the model simply learned this as a shortcut. Models trained for Polyp-segmentation and detection are also subject to generalization failure, as evidenced by EndoCV2021 [1, 3]. Even the best-performing models exhibited considerable performance degradation when evaluated on unseen datasets, collected from separate centers or with differing lighting modalities.

### 2.3.2 Generalization Failure in General

Though generalization failure is perhaps best represented in medical domains, the phenomenon is pervasive in practically every application of deep learning, albeit to varying degrees. It has for instance been shown that CNNs trained on ImageNet, one of the largest and most diverse datasets in the domain of computer vision, are heavily biased towards textural features, and consequently fail when the texture of the input is modified, despite the shape and structure of the relevant object remaining recognizable [28]. Though this result is based on evaluation on synthetic data, it highlights a key property of deep learning pipelines: namely that they do not necessarily learn features that are causal - in other words, that they are intrinsic to the relevant object - inasmuch as they learn features that are highly correlated with it - in other words, features that are associated with the object but are not intrinsic to it. Though the texture of cat fur for instance is highly correlated with the "cat" class, it is not the fur that makes the cat. In Figure 2.6, for instance, it is clear that image (c) should be classified as a cat more than an elephant. Granted, this example is as mentioned synthetic, but a similar situation could arise if the classifier for instance was tested on a black-and-white image of a hairless cat.

This behaviour of considering correlations over causation can also be found in state-of-the-art image captioning systems, for instance Microsoft Azure's computer vision API and NeuralTalk2 [83], wherein the model seemingly hallucinates that it sees sheep when evaluated on images of grassy pastures or hills. This is shown in Figure 2.7. Once again, it is of course natural to expect that sheep can be found in these contexts, but it is not these contexts


Figure 2.6: Classifiers trained on ImageNet are biased towards textural features. Adapted from [28] under Creative Commons 4.0.

that define what it means to be a sheep. Grassy pastures and sheep are not causally related, only correlated, but deep learning pipelines lack the nuance required to understand this fact.


Figure 2.7: Deep captioning models hallucinate sheep (and other animals) when presented with contexts highly correlated with sheep. Adapted from [83]

Another characteristic of deep learning that supports this argument is the effectiveness of adversarial attacks [44], which specifically target weaknesses in the representations used by a given DNN through any number of means in an attempt to induce high rates of incorrect, yet highly confident predictions. Gradient-based adversarial attacks, for instance, use the gradients of the model to break even the most sophisticated and well-

trained pipelines merely by adding some carefully crafted, yet visually imperceptible noise to the inputs [13]. Even without access to the gradients, there exists a multitude of so-called black-box attacks that only use output samples to generate similarly effective attacks [45]. Finally, it has been shown that adding minor visual distractions to objects, for example adding bits of tape or graffiti to stop signs, dramatically increases misclassification rates [26].

## 2.4 Generalizability Theory

Exactly why and how DNNs seem to so persistently fail to generalize is a topic of ongoing research. The available literature is fairly fragmented, often making use of differing and sometimes conflicting terminology. Moreover, the literature suggests that generalization failure is a highly multifaceted problem, with many potentially affective variables. This section will summarize the analyses performed throughout the literature, and attempt to distill them in a manner more conducive to the development of generalizable methods. The section will start by discussing the theoretical basis for why one might expect DNNs to generalize, discuss the key characteristics of generalization failure, and finally discuss why and how these characteristics arise according to analyses in the literature.

### 2.4.1 Generalization through Empirical Risk Minimization

Deep learning would not be as ubiquitous as it is if there was not some semblance of an expectation that their striking performance could generalize to outside the idealized settings typically involved in research. The theoretical basis that informs this belief in (most) modern deep learning pipelines is the idea of so-called Empirical Risk Minimization (ERM). Thus, to fully understand why generalization failure occurs, it is beneficial to analyze ERM from first principles:

At the most fundamental level, the goal of machine learning is to learn a mapping between two spaces of objects $X$ and $Y$. This mapping, namely the function $f : X \rightarrow Y$, maps some input object $x \in X$, an image for example, to a corresponding and application-relevant output object $y \in Y$, for instance a segmentation mask or class-wise probabilities. It is worth noting, however, that $f$ is not as much a function in the mathematical sense as much as it is an abstraction of the relationship that the deep learning system is intended to capture. $f$ cannot as a consequence typically be modelled explicitly. Instead, machine learning systems aim to find a sufficient approximation of this mapping by leveraging a training set $\{x_i, y_i\}_{0...n}$. This is referred to as *supervised learning*, and the resulting approximation found using the training set is denoted by $h : X \rightarrow \hat{Y}$, and

typically referred to as the *hypothesis*.

To find such an approximation, it is assumed that there exists a joint probability distribution over $X$ and $Y$, namely $P(x, y)$, and that the training data $\{x_i, y_i\}_{0...n}$ is drawn from this probability distribution such that the resulting sample distribution Independent and Identically Distributed (IID) to $P(x, y)$. This is referred to as the IID assumption. By modelling the mapping as a joint probability distribution, one can model uncertainty in the predictions by expressing the output as a conditional probability, $P(y|x)$. In conjunction with a loss-function $L(h(x), y)$ which measures the discrepancy between the hypothesis and the ground truth, this allows us to quantify the expected performance of a given hypothesis:

$$R(h) = \mathbb{E}[L(h(x), y)] = \int L(h(x), y) dP(x, y) \tag{2.5}$$

Using this framework, one can then find an IID-optimal hypothesis, often called a *predictor*, by finding the predictor $h^*$ among a fixed class of functions (defined by network architecture) $\mathcal{H}$ that minimizes risk:

$$h^* = \arg\min_{h \in \mathcal{H}} R(h) \tag{2.6}$$

Since $P(x, y)$ is not known, however, one cannot compute $R(h)$ explicitly. Instead, the expected risk has to be estimated empirically, i.e by finding the arithmetic average of the risk associated with each prediction by the hypothesis over the training set:

$$R_{emp}(h) = \frac{1}{n} \sum_{i=1}^{n} L(h(x_i), y_i) \tag{2.7}$$

This risk can in turn be minimized with respect to the hypothesis class. This is called empirical risk minimization (ERM):

$$\hat{h} = \arg\min_{h \in \mathcal{H}} R_{emp}(h) \tag{2.8}$$

To reiterate, the central idea with this approach to machine learning is that the training data can be considered a finite IID sampling of the underlying distribution. As such, by the central limit theorem, the hold-out performance of the computed hypothesis will approach IID-optimal performance given a sufficient amount of training data and some sufficiently capable training procedure. This should in theory allow deep learning systems to be able to generalize, since the empirical risk in theory can approximate the true risk arbitrarily well given sufficient training data.

As described in Section 2.3, ERM nonetheless readily fails to generate generalizable predictors with respect to out-of-distribution data. There are multiple dimensions to this phenomenon, as there are several means by which a model can fail to generalize. To better understand these failure modes, it helps to state the assumptions that are made in the formulation of ERM, namely that:

1. $f$ exists in $\mathcal{H}$

2. The optimal predictor can be found solely through minimizing (The IID assumption) $R_{emp}(h)$

3. $\{x_i, y_i\}$ is an IID sampling of $P(x, y)$

4. $\hat{h}^*$ is unique in $\mathcal{H}$

As the following sections will show, violations of any one of these assumptions can and typically will result in generalization failure.

### 2.4.2 Realizability and Underfitting

Violations of assumption 1 corresponds to a well known and fairly well understood form of generalization failure, namely underfitting. One can however argue that underfitting can be all but discounted as plausible explanation for the pervasiveness of generalization failure observed in modern deep learning pipelines. Underfitting occurs when the model simply lacks the complexity required to encapsulate the patterns necessary to form generalizable interpretations of the data. To give a simple example - consider the problem of trying to fit a linear model to a dataset wherein the variables are related by a quadratic function, e.g $y = x^2$ as shown in Figure 2.8. No amount of optimization of the parameters in the linear model can ever result in a sufficient description of the underlying data and the function the constituent variables are related by.

This, however, does not necessarily mean that an underfitted model cannot perform well; the data shown in Figure 2.8 function is after all locally linear, and if it is only evaluated on a limited region, a linear model may perform sufficiently. One can as such argue that DNNs in turn may be underfitting, and that generalization failure analogously corresponds to evaluating on data outside of this locally linear region. This, however, is unlikely to be the case, as evidenced by recent results in the study of model complexity.

Modern DNNs, as it turns out, have practically infinite *effective capacity* - i.e., they can model more or less arbitrarily complex data. It can for instance be shown that even a 2-layer feedforward neural network is capable of fitting noise to random labels with 100% accuracy [100] so long as it is sufficiently wide. Consequently, it is fairly reasonable to expect that the hypothesis space of the highly complex models used today contains a generalizable predictor and thus that assumption 1 holds. In the literature, this is often referred to as the realizability assumption [82].

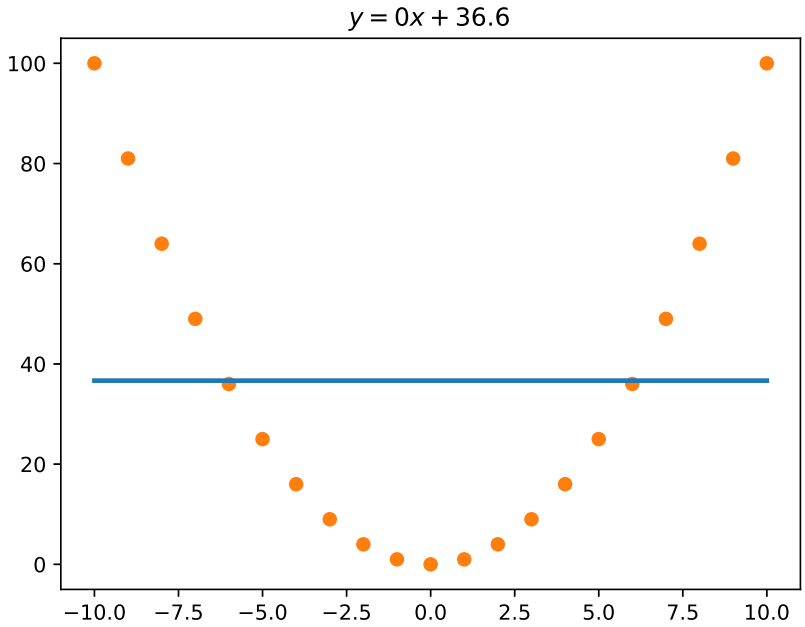


Figure 2.8: Example of a linear model underfitting polynomial data. The residuals are in this case minimized, but the model is nevertheless only correct near two datapoints.

### 2.4.3 Overfitting, Inductive biases and training

The high effective capacity of DNNs does, however, result in a number of side-effects that actually hamper generalization. Though this capacity does suggest that most learning problems are realizable, the problem of finding a generalizable predictor from the hypothesis space is nevertheless not at all trivial. ERM presupposes assumption 2 - i.e that there exists some way to precisely find the risk-minimizing predictor $\hat{h} = \arg\min_{h \in \mathcal{H}} R_{emp}(h)$, and as such that there is some ideal optimization procedure that can be leveraged to this end. This, of course, is not the case. Instead, a search of the hypothesis-space is performed using gradient-descent. On its own, this search is not necessarily guaranteed - or for that matter even likely - to find an IID-optimal predictor. This is due to the inherent nature of the search space - DNNs have parameter counts numbering in the millions or more, and try to determine optimal parameter configurations from comparatively miniscule datasets.

Without certain precautions, this may result in the pipeline returning predictors that in effect simply memorize the training data, without learning anything useful about the domain itself. This is referred to as overfitting [32].

Memorizing all the training data is, however, risk-minimizing. To illustrate, consider a predictor which simply memorizes the segmentation masks for the polyps in a given dataset, and simply returns the corresponding mask when given an image it has trained on, and returns a zero-mask otherwise. This, as explained earlier in this section, is entirely within the capabilities of DNNs due to their high effective capacity. When evaluating this predictor on the dataset upon which it was trained, the empirical risk will be zero since it will correctly return the right segmentation for a given image despite not having learned anything useful about polyps whatsoever, or for that matter anything useful about images.

Thus, certain constraints have to be imposed on the search space to avoid overfitting. These constraints have to be defined a-priori, and are often referred to as the *inductive biases* of the pipeline.

This is often achieved through the use of regularization techniques. Dropout [40], for instance, biases the model towards learning representations that distribute well across the network and can work independently of one another. Weight decay [55] biases the model toward low-magnitude parameters, and thus in theory simpler representations. Data augmentation biases the model towards learning features that hold across augmentations, and so on.

Besides regularization, certain inductive biases can also be imposed through modifying the training routines themselves, by for instance through *pretraining* [25] - i.e first training the model on more general or otherwise related data, *contrastive representation learning* [60] - i.e. learning to represent similar samples coherently in an unsupervised manner - or *multi-task learning* [78, 89] - i.e. learning representative features through multiple tasks.

Determining the effectiveness of these techniques and tuning the hyperparameters that inevitably arise also requires a specific evaluation procedure. To this end, most deep learning pipelines leverage *hold-out sets*, wherein the data is partitioned into three folds - the *training set*, used to compute gradients and train the model, the *validation set*, used to tune hyperparameters, and a *test-set*, used to evaluate the performance of the model [32]. More sophisticated methods, such as *cross-validation*, are also often used. Note that evaluation on test-sets only determines the generalization to data that is IID to the training set, also typically referred to as In-Distribution (InD) data.

Fundamentally, each of these techniques increase generalization by limiting the search space, in effect redefining $\mathcal{H}$. The more inductive biases are imposed onto the model, the smaller $\mathcal{H}$ in effect will be.

Modern deep learning pipelines regularly employ several of these techniques, often in conjunction with one another, and consequently easily avoid overfitting and achieve good results on the test-set. This only guar-

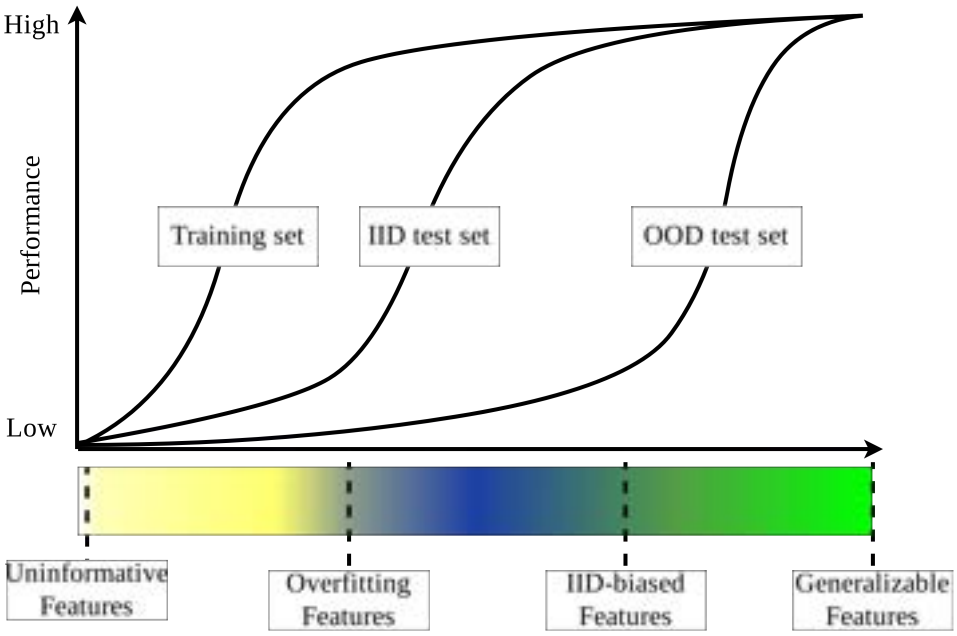


Figure 2.9: Good performance on unseen InD test sets do not guarantee generalization, as it only requires learning InD-biased features. OOD-suitable performance requires that the model learns causally related and thus generalizable features. Adapted from [29]

antees InD generalization, however, and thus these models still readily fail to generalize to OOD data. This is illustrated in Figure 2.9 That is not to say that regularization and other ways of imposing inductive biases on the model does not aid in generalization, only that overfitting does not explain the pervasiveness of generalization failure that can be seen today.

### 2.4.4 Structural Misalignment and Dataset Bias

Recent research attributes generalization failure to *structural misalignment* between the features that predictors learn through ERM and the causal structure which they ideally should encode [5, 29, 44, 81]. Generally, this misalignment occurs as a result of the predictor learning spurious or otherwise causally unrepresentative features that nonetheless perform well within the training distribution. This if of course made evident as soon as the predictor is exposed to any form of distributional shift, at which point it will fail to generalize. These distributional shifts can range in magnitude, from changes in imaging modalities [1, 20], common corruptions such as noise or blurs [37], or spatial transforms [23], to practically imperceptible perturbations, typically exemplified by adversarial attacks [13]. ERM does not and cannot guarantee invariance to distributional shifts, as it assumes that the training data is IID to $P(x, y)$. This is not, however, necessarily as much of a flaw with ERM inasmuch as it is a flaw in the reasoning behind

our expectations.

To illustrate, consider the rather pertinent example of training a model exclusively on either white-light or narrow-band endoscopy. Assume that there are two datasets, each containing samples depicting identical scenes, with the only difference being that dataset A employs white-light endoscopy, whereas dataset B employs narrow-band endoscopy. Ideally, a model trained on either dataset should generate predictors that can generalize to the other, but this is in no way guaranteed. The causal structure behind the problem - i.e. what exactly makes a polyp a polyp - is never considered at any point in the training process. Instead, the models will simply try to leverage any arbitrary predictive pattern that can be found the training data. The model trained on narrow-band images may for instance principally consider the textural characteristics of the polyps, which narrow-band endoscopy enhances. Conversely, the model trained on white-light images, lacking access to these textural characteristics, may instead be biased towards more color- or shape-based features. If this narrow-band-texture-biased model is deployed in white-light endoscopy, it is not likely to succeed since its principal discriminative features no longer are particularly useful. Similarly, the color-biased model would likely fail when deployed in narrowband endoscopy since the colors it once used to distinguish polyps would no longer predictive.

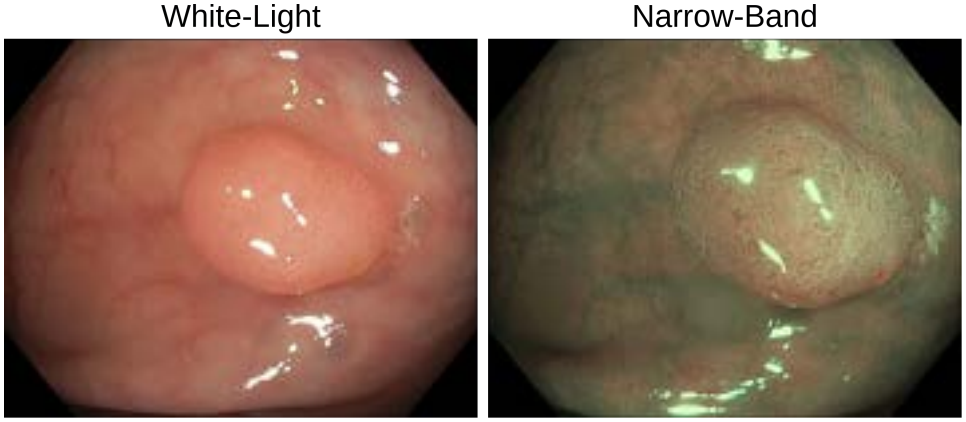


Figure 2.10: The difference between white-light endoscopy (left) and narrow-band endoscopy (right) constitutes a distributional shift. Models trained with ERM on one of these modalities cannot be expected to generalize to the other.

Though the features each model learns are not particularly representative of the broader context of what makes a polyp a polyp, they make sense when considered from the perspective of either of the two modalities. When considering only narrow-band imaging, it makes some sense to heavily weigh the texture of the polyps. When considering only white-light imaging, it makes some sense to heavily weight the shape and color of the polyps. Though humans are capable of appreciating broader context and subconsciously know that certain features are ancillary rather than causal

(and perhaps more importantly: know the strengths and weaknesses of each modality), DNNs lack the inductive biases needed to take this into account. Once again, DNNs merely leverage the first and best predictive patterns found during the training process, and cannot be expected to optimize for specific invariances on their own, irrespective of how self-evident these invariances may be for humans. This predilection towards dataset-specific features is aptly referred to as dataset bias.

### 2.4.5 Shortcut Learning

In the example introduced in the previous section, it was assumed that a model trained on datasets consisting of images of a given modality - e.g narrow-band endoscopy - would learn features that correspond to causal relationship within that modality. I.e. for narrow-band endoscopy, polyps are (in part) defined by their specific textural characteristics. Thus, though this relationship is not dataset-agnostic, it is at causally viable and would generalize so long as it was deployed exclusively in narrow-band endoscopy.

As it turns out, however, CNNs are unlikely to learn such causally viable features in the first place. In other words, the predictors would not necessarily learn to consider texture in narrow-band images - it could learn any arbitrary pattern so long as it is predictive. Moreover, if such interpretable distributional shifts were the principal cause of generalization failure, generalizability could be practically guaranteed by explicitly modelling the effects such shifts induce and taking these into account when training models. In the aforementioned example, one could for instance train some model to map from one lighting modality to the other. Though this would imbue the model with an inherent invariance to the choice of lighting, it is nonetheless not certain that the resulting model would be perfectly generalizable.

Consequently, though these detectable forms of distributional shifts also hold some importance when designing generalizable models, a more pervasive and substantially more significant issue is the fact that many of the distributional shifts encountered in clinical settings are not necessarily considered significant or for that matter at all perceptible to a human observer. A human would for instance not be significantly affected by slightly noisy, blurry, rotated, or compressed images, nor would they in all likelihood even notice these perturbations. DNNs, on the other hand, have been shown to be highly sensitive to these and several other forms of minor perturbations [37, 42, 85].

Moreover, though a human would likely not pick up on subtle phenotypic cues that may exist in the colon during endoscopy, whereas a DNN may leverage some of these cues to inform their decisions, and hence exhibit varying performance across different demographics.

It is important to note, however, that despite how these two forms of distributional shift may at surface level appear as completely separate classes of problems, they can both be traced to the same root cause - namely that DNNs do not leverage any form of causal logic to inform their decisions and, as mentioned previously, simply exploit any sufficiently predictive pattern they may observe in the data. This is often referred to as *shortcut learning* [29] or *the Clever Hans effect* [51].

Shortcut learning and the resulting brittleness of the features that it induces has been identified as one of the key phenomena that explain the effectiveness and pervasiveness of adversarial attacks [44]. Adversarial attacks simply leverage the high degrees of sensitivity inherent to shortcut features, and construct perturbations according the direction in the search space that corresponds to the principal component of this sensitivity [68]. A generalizable predictor should be robust to such minor perturbations, as the model should not in the first place be learning features that get perturbed to any significant degree by adding high-frequency, low-amplitude noise.

### 2.4.6 Underspecification

Closely related to shortcut learning is underspecification [20]. A machine learning pipeline can be considered underspecified when it can return any number of risk-equivalent predictors when evaluated on an InD holdout set, dependent only on the random variables used within the training procedure - i.e dropout, seed initialization, and so on. Even with identical hyperparameters, a given training procedure can return any number of predictors each having learned different patterns - be it shortcut features or causal features. One predictor may have learned one shortcut, another may have learned an entirely different but nonetheless equally predictive shortcut, and one may have fully learned the actual causal relationships it is intended to. With ERM, and in particular with InD-oriented evaluation procedures, these are all erroneously considered equivalent.

This is evidenced by the significant variability in performance that can be observed when testing several predictors that are identical except for the choice of random seed on OOD datasets [20]. Typically, this variability is orders of magnitude larger than the variability the same group of models exhibit on an InD test set.

### 2.4.7 A probabilistic perspective of generalization

As established, modern deep learning pipelines are not capable of reliably returning generalizable predictors. However, they are not necessarily precluded from it. One can to some extent model this probabilistically by

considering the distribution of parameters given the training data, $p(w|\mathcal{D})$. Though it is impossible to know which part of this distribution corresponds to generalizable predictors, it has been shown that marginalizing over this distribution increases generalizability [8, 41, 88, 94]. This is referred to as Bayesian Learning [93].

The details and statistical nuances behind this is somewhat outside of the scope of this thesis, so for now it is sufficient to simply consider it as a way to account for some of the variability inherent to the distribution of predictors that can be generated by a deep learning pipeline.

This view can be more readily understood by taking a probabilistic perspective of generalization as a whole [94]. Generalizability can be considered as a two-dimensional quantity, consisting of the support and inductive biases of a model. The inductive biases are as mentioned the constraints by which the model learns, which for instance can be induces through the design of the model-architectures - e.g positional invariance in CNNs, regularization - e.g. dropout, data augmentation, etc., or specific training routines - e.g. multitask learning, contrastive representation learning, and pretraining. The support, on the other hand, describes the ability of the model to encode certain decision rules. Following the Bayesian perspective, both the support and inductive biases should be maximized. Maximal support permits a given model to learn arbitrarily however complicated decision rules are required for a given task, and a maximal set of inductive biases reduces the probability of learning decision rules that, though predictive, are not causally related to the problem.

## 2.5 Related work on Generalizable Deep Learning

To summarize the preceding sections, generalization failure occurs due to the weaknesses inherent to ERM. The features that predictors trained with ERM learn to incorporate are often spurious, and the deep learning pipeline can return any number of spurious or non-spurious predictors from identical training procedures up to choice of random seeds. The approaches that have exhibited the highest degrees of success towards increasing generalization as a consequence tend to address these issues in some way or another. This section will discuss a number of such methods, including data-augmentation, model-debiasing, alternative learning paradigms, and ensemble models.

### 2.5.1 Data-augmentation

One of the most well-studied approaches to increasing generalizability is the use of data augmentation. Data augmentation is typically implemented

in deep learning pipelines in order to prevent overfitting, often in conjunction with other regularization methods. As discussed earlier, overfitting constitutes generalizability failure in its own right, but augmentation has also been shown to have positive effects for out-of-distribution generalization [31]. It has for instance been shown that carefully designing augmentation procedures increases the generalizability of polyp segmentation models [30] and prostate segmentation models [80].

Data augmentation can be interpreted as providing a better estimate of the overall risk $R(h)$. This is because the empirical risk will be best minimized by leveraging features that are conducive to minimizing risk across both the augmented data and unaugmented data. Indeed, it has been shown that using data augmentation has comparable effects to incorporating a second, OOD dataset as additional training data [31].

The effects of augmentation can be understood from the perspective of generalization as mentioned in Section 2.4.7. In effect, data augmentation is simply a method by which additional inductive biases can be imposed. This increases the likelihood of learning generalizable features. For instance, by augmenting with random rotations, rotational invariance is presupposed. By augmenting with color-jitter, invariance to global color-transforms is presupposed. By employing additive noise, invariance to additive noise is presupposed, and so on. There has also been a large body of work dedicated to leveraging recent advances in generative models such as Generative Adversarial Networks (GANs) and Variational AutoEncoders (VAEs) to serve as synthetic data augmentation. These types of approaches have also been shown to increase generalizability, in particular in medical domains such as CT segmentation [79] and x-ray based detection of covid-19 [67]. To understand why this is the case, and as the methods proposed in Chapter 3 leverage a form of GAN, it is worth explaining how these networks work.

#### Generative Adversarial Networks

GANs are very simply put a type of deep learning framework intended to be able to replicate the distribution upon which they are trained. A GAN trained on images of human faces, for instance, would in theory generate a practically infinite number of novel, believable human faces [50].

This is achieved through a particular training regiment, wherein two models - the generator and the discriminator - contest one another in a zero-sum game. This is shown in Figure 2.11. The generator, as the name suggests, attempts to fool the discriminator by generating synthetic data from noise, intended to be indistinguishable from what one might expect a sample from the training distribution to look like. The discriminator, on the other hand, tries to classify an incoming image - either from the dataset or from the generator - as synthetic or genuine. This is achieved by

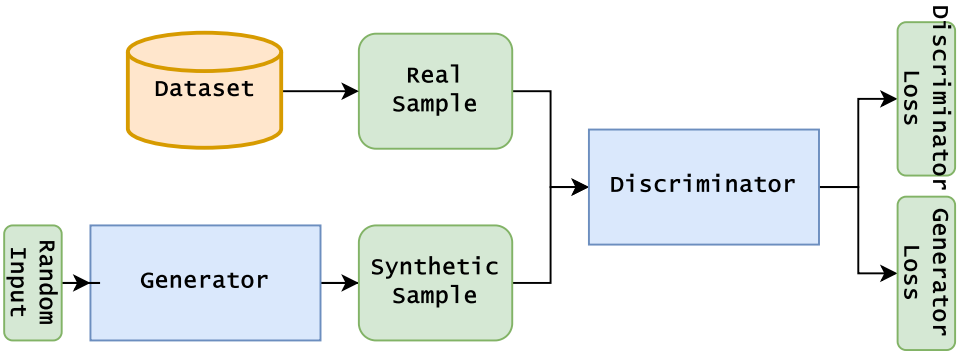


Figure 2.11: Diagram showing the regiment for training GANs. The generator produces a synthetic image with noise as input, which the discriminator then tries to classify as real or fake along with real examples.

training the generator and discriminator in an interleaved fashion, wherein the gradient updates for each model are a function of the output of its adversary. To illustrate, consider a basic GAN pipeline [33]: Let $G(\cdot)$ be the generator, and $D(\cdot)$ be the discriminator. Let $x$ be a training input sample, and $z$ be random noise. The generator will then try to minimize the following quantity, whereas the discriminator will try to maximize it:

$$\mathcal{L}_G = \log D(x) + \log\left(1 - D(G(z))\right) \tag{2.9}$$

This way, when the generator is being trained, it learns to generate outputs that fool the discriminator. This corresponds to the discriminator outputting a high probability of the image being real, i.e $D(G(z)) = 1$. Conversely, when the discriminator is being trained, it learns to generate outputs that correctly classify the generated samples as fake, i.e $D(G(z)) = 0$, and the real samples as real, i.e $D(x) = 0$.

#### GANs, Generalization and Modelling the Distribution

Ideally, the fully trained generator should be capable of generating the full space of images defined by the training distribution $\mathcal{X}$ simply by modulating the input vector. And indeed, mathematical analysis shows that GANs are capable of approximating this distribution arbitrarily close given infinitely sized datasets, infinite training time and infinite model support [33].

Whether this is the case in practice is a matter of ongoing research, however, with mathematical analyses suggesting that sufficient approximation of the distribution is impossible without the aforementioned assumptions [6].

This is evidenced by the pervasiveness of a phenomenon referred to as mode-collapse, wherein GANs learn to replicate only a limited subset of

distribution. One can argue that this in effect stems from GANs failing to generalize [7]. This limits the potential of GANs somewhat; after all, if they really did model the distribution, one could leverage GANs to generate practically infinite synthetic datasets and thus train highly generalizable models.

That is not to say, however, that GANs lack utility as an augmentation method. As mentioned previously, GAN-based data augmentation techniques have been shown to have the potential to increase the generalization of the target models. This is, however, not typically achieved merely through training on synthetic images, but by training GANs such as CycleGAN [79] or other distributional models [67] to translate between domains.

#### Gan-Inpainters

Of particular interest in the context of segmentation is also GAN-inpainters, which as the name suggests fill in masked regions in an image with pixels such that the resulting scene is maximally believable [71]. This is achieved by using a configuration similar to what is shown in Figure 2.11, but with some modifications:

First, the generator has to train to optimize for two objectives: fooling the discriminator, and minimizing the pixel-wise distance between the inpainted regions and the true regions. Second, the discriminator has to learn to classify pixels as either being inpainted or real. This results in the following optimization objectives, both of which are minimized:

$$L_g = \lambda_1 BCE(D(x), y = 1) + \lambda_2 L1(G(x), x) \tag{2.10}$$

$$L_d = \frac{1}{2}[BCE(D(G(x), y = 1) + BCE(D(G(x), y = 0))] \tag{2.11}$$

Where the $\lambda$ terms correspond to weights, treated as hyperparameters.

As will be explored in Chapter 3, this can be used to augment segmentation tasks by training the model to inpaint the segmentation target class only, and thus add additional regions corresponding to the given segmentation target to an image. In this thesis, this will involve inpainting additional polyps in a given endoscopic image.

### 2.5.2 Model Debiasing

Another type of approach involves biasing the pipeline towards learning more structured and causally viable latent representations - or, equivalently, debiasing it from learning spurious correlations [31]. This is also

somewhat well understood when considered through the lens of regularization: dropout [40] and weight-decay [55] are often employed in order to reduce overfitting under the assumption that a generalizable predictor should not base its decisions on only a few of the available weights, and that separate components in the networks should instead encode independent representations of the input. Though there is limited research on the effects of regularization methods other than data augmentation on OOD generalization specifically, debiasing through constraining the space of latent representations that a DNNs can leverage has been shown to be effective method of increasing generalizability. In the case of polyp-segmentation it has for instance been shown that adding context-based attention layers to multiple blocks to a network results in a significant increase to OOD performance [53].

Multi-task and multi-stage learning has also been leveraged for the purpose of model debiasing. By jointly optimizing for multiple tasks/subtasks, the model can be biased towards learning features that describe the input data well independent of their performance on any one of the relevant tasks. For polyp-segmentation, for instance, it has been shown that adding image reconstruction as an auxiliary task in conjunction with attention-blocks [89] or decoupling the segmentation task into a coarse segmentation and refinement stage [27] increases generalizability.

More closely supervised methods, wherein certain inductive biases are introduced to the pipeline in a more explicit manner, have also been shown to have some promise. One paper for instance reported increased robustness to image perturbations after adding a custom filter bank designed to emulate the primary visual cortex of primates to the front of a CNN [21]. Another paper reported that models trained on Imagenet exhibited significantly higher robustness when explicitly biased towards shape-based features [28].

### 2.5.3 Novel Learning Paradigms

A growing body of work has also investigated the idea of foregoing ERM altogether, or at least certain elements thereof, in favor of developing alternative training paradigms.

In so-called Invariant Risk Minimization [5], for instance, the model trains to ignore spurious correlations by optimizing for predictors that exhibit stable performance across several datasets.

Model-Based Robust Deep Learning [75] employs a similar idea in conjunction with distributional modelling. The model is trained such that it learns invariance to perturbations as applied by a generative model trained to map inputs continuously between separate domains in accordance with a nuisance parameter. If this model for instance

describes the function mapping white-light endoscopy images to narrow-band endoscopy images, this will then optimize for predictors that leverage features that generalize to both lighting methods and any combination thereof.

It should be noted, however, that these methods all necessitate multiple datasets, and may as such have limited utility in domains where datasets are scarce.

### 2.5.4 Bayesian Marginalization and Ensembles

Finally, So-called ensemble networks have demonstrated high degrees of generalizability for polyp-segmentation, and account for three of the eight accepted submissions to the segmentation task of EndoCV2021 [41, 57, 88]. An ensemble model is in effect a set of distinct predictors which generates outputs according the consensus of its constituents. This requires training multiple models independent of one another, and works under the assumption that considering multiple representations of the data concurrently facilitates increased generalization.

It can be shown mathematically that this ensembles an approximation of Bayesian marginalization [93, 94]. One can consider an ensemble to be a sampling of the Bayesian posterior - i.e., $p(w|\mathcal{D})$. Consequently, ensemble-based networks can mitigate underspecification to a certain extent, merely by representing a higher proportion of the space of possible predictors, thus increasing generalizability. As each predictor is unlikely to have learned identical representations, any spurious correlations inferred by one predictor will not affect the final prediction so long as they have not been learned by the majority of the predictors in the ensemble.

There is, however, a caveat to this. It may for instance be the case that a given pipeline returns predictors that have learned one specific shortcut (or a set of shortcuts) in the majority of runs, in which case no amount of Bayesian marginalization or use of ensemble models will ever result in appreciable increases to generalization. This is illustrated in Figure 2.12. Nevertheless, the literature has demonstrated the generalizability of ensemble models, though the relative impacts thereof is still to some extent poorly understood.

## 2.6 Ethical considerations

Now that generalization failure and the prevalence thereof has been discussed in quite some detail, as well as the limited success of the literature with regards to resolving this problem, it is beneficial to pause to consider the ethical and social consequences inherent to the it.

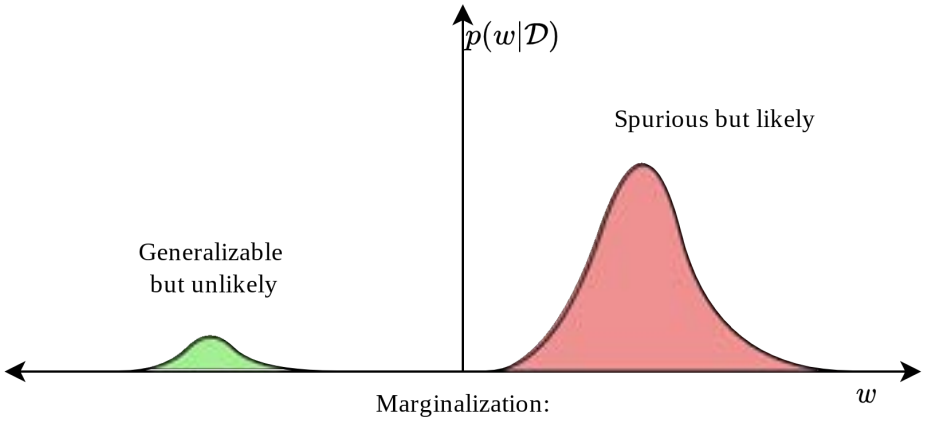


Figure 2.12: Bayesian marginalization may not yield generalizable predictors. In the above case, the portion of the distribution that correspond to non-generalizable features (red) are more likely to be learned than generalizable features (green). Spurious features will as a consequence constitute the principal component of the marginalized probability, in turn resulting in minimal impact on generalization.

The robustness and generalizability of medical systems is of exceptional importance [11], and the possible negative consequences for incorrectness or even simply unreliable performance are significant. Over-reliance on a non-robust deep learning model may for instance result in increased miss rates in the case of polyp screening, which of course may have fatal consequences should the polyp turn cancerous. Clinicians may grow accustomed to a high-performing model and then perhaps more readily fail to notice should it leave polyps undetected if for instance the distribution shifts such that this performance is hampered. In this case, the addition of the deep-learning based screening method would in fact do more harm than good.

As briefly mentioned in Chapter 1, sampling bias in the datasets upon which these models are trained may also lead to inequality of treatment, for instance if certain demographics are not accounted for [69]. Though this of course can be mitigated to some extent by more careful curation of datasets, the central problem that results in this disparity is at its core the fact that the model is learning spurious features. A model intended to detect melanomas should not, for instance, rely solely on complexion-dependent characteristics to make predictions. Though curating datasets with a diverse representation of skin-tones would mitigate this problem to some extent, it is uncertain, and as the analysis in previous sections shows unlikely that all potential variability would be accounted for. Thus, ensuring that a given model learns causally robust and thus generalizable

features is of particular importance also in this regard.

## 2.7 Summary

This section has covered the basics of deep learning and segmentation, discussed a number of documented cases of generalization failure, and summarized a number of analyses of generalization performed in the literature.

Generalization failure is prevalent across practically all deep learning pipelines. The mechanisms behind these failures are only loosely understood, and there has been limited success in the endeavor of developing generalizable methods to increase the generalizability of deep learning pipelines. Generalization failure can in broad strokes be attributed to the inability of empirical risk minimization to consistently learn causal patterns, and that predictors trained with ERM instead favor whatever patterns that can be found in that are sufficiently predictive in a InD context. Methods that address this in some way - for instance ensembles, data augmentation, etc - consequently tend to increase generalizability.

The relative impact of these methods is, however, poorly understood, and the literature is to some extent fragmented with regards to the experimental methodology used when evaluating the generalizability of models. Moreover, there have been limited efforts made towards developing novel approaches explicitly aiming to increase generalization without relying on auxiliary OOD datasets in the training process. Hence, addressing this issues is the primary focus of this thesis. The next chapter explores the impact of several novel methods to this end, including the use of a dual-decoder model for multi-task learning, a novel augmentation strategy which includes a generative inpainter, a novel training procedure referred to as Consistency Training, and finally a number of ensemble models trained using Consistency Training. These will then be compared to corresponding baselines in Chapter 4, which in turn will be compared to one another in order to ascertain the relative impacts of model architectures, data augmentation, and the use of ensemble models on generalization.

# Chapter 3

# Methods

Summarizing the key points made in Chapter 2, current deep learning pipelines are not equipped with evaluation methods suitable for determining the degree to which predictors can generalize to OOD data, are prone to learning spurious features, and are underspecified by the datasets they are trained on. These factors can be traced back to shortcomings in Empirical Risk Minimization (ERM), the theoretical basis for deep learning. The literature around developing methods to address these shortcomings tends to focus on developing more generalizable model architectures [27, 35, 36], data augmentation [30, 34, 79], Bayesian marginalization through ensembles [41, 57, 88], or developing novel training paradigms to directly work around the shortcomings of ERM, typically by incorporating multiple training domains [5, 75].

Restating the research objectives, this thesis aims both to determine the relative impacts of a number of these methods, which will be considered further in Chapter 4, and to develop novel methods as informed by the theory presented in Section 2.4. To this end, this Chapter will introduce the following methods:

- A modified DeepLabV3+ model with dual decoders, intended to constrain the space of latent representations such that underspecification is mitigated.
- A novel framework for analyzing generalizability based on reframing it as a predictor's ability to exhibit consistent behaviour with respect to distributional shifts, as well as a corresponding training procedure, metric and loss-function. This is in effect an alternative to data augmentation, and in contrast to competing methods [5, 75] does not make use of OOD datasets.
- An augmentation strategy informed by this alternative view, including both conventional augmentations and a GAN-inpainter.

- A family of ensemble models consisting of predictors trained according to the above methods.

This chapter will detail the development of these methods, including their basis with respect to the theory as outlined in Section 2.4.

## 3.1 DD-DeepLabV3+

As described in Chapter 2, generalization failure can in part be attributed to the fact that most deep learning models are underspecified by the training data. In other words, the same pipeline can return any number of risk-equivalent predictors that leverage significantly different features. To mitigate this, one may debias the pipeline by imposing constraints on the space of features that a given model can learn, such as through multi-task learning [89], attention-mechanisms [35, 98] or preprocessing [21].

As testing the relative effectiveness of all of these different approaches is beyond the scope of this thesis, only a simple dual-decoder model is introduced, namely DD-DeepLabV3+. As its name suggests, this model is functionally equivalent to a standard DeepLabV3+ [18], but is endowed with an additional decoder, which performs image reconstruction. In theory, this should constrain the model such that it learns features that are conducive to both segmentation and reconstruction simultaneously. This constraint should mitigate underspecification and force the model to learn more generalizable features, as the feature space that is conducive to both reconstruction and segmentation should be smaller than the feature space conducive to segmentation only. A diagram of the model is shown in Figure 3.1.

This model also has the advantage of being easily compared to the standard DeepLabV3+; the part of the dual-decoder network responsible for segmentation is after all functionally identical to the single-decoder network. This facilitates better analysis of the impact of the additional decoder and its effect on the learned features, as the performance of the respective models can be compared directly.

## 3.2 Consistency Training

This section will introduce Consistency Training, illustrated in Figure 3.2, a training procedure wherein the intent is to optimize for generalizable features by minimizing the degree to which the model outputs inconsistent predictions when the input is subjected to a set of transformations. This is achieved by training with two versions of each batch: one which

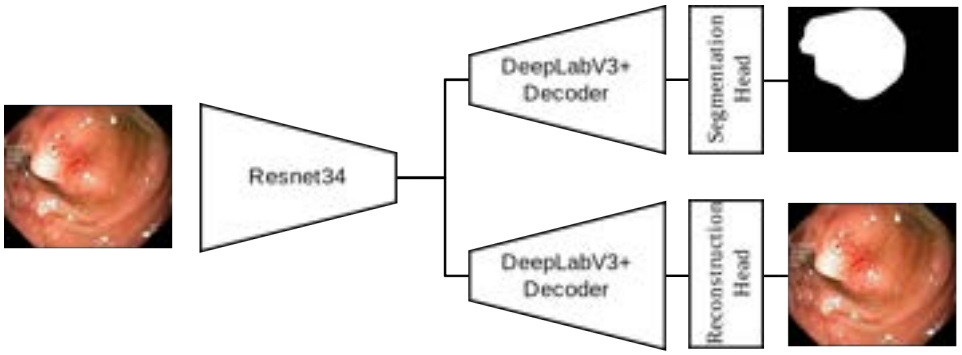


Figure 3.1: Diagram showing the Dual-decoder DeepLabV3+ model. This model uses a ResNet34 encoder to generate a feature map, which is then leveraged by two decoders concurrently. One decoder performs polyp segmentation, and the other performs image reconstruction. Functionally, the decoders are identical, and differ only in that the segmentation decoder requires sigmoid activation to map the output logits to a probability map one channel wide, whereas the reconstruction maps the logits to RGB values.

is augmented, and one which is not. The given model then performs inference on these two images, resulting in two segmentation masks. The difference between these two predictions is then computed, and compared to the difference (if any) between the augmented and unaugmented segmentation labels. This is then incorporated as a loss such that the discrepancy between the expected prediction change and actual prediction change is minimized.

One can draw parallels between this pipeline and contrastive learning [60], which also makes use of a similarity metric as computed from separate outputs of the same model. However, whereas contrastive learning is primarily used in unsupervised settings, often as pretraining, Consistency Training is in effect instead a more generalizable alternative to data augmentation.

The next sections will cover the theoretical basis of this training procedure as well as the implementation of its constituent components, therein a novel loss function, augmentation strategy, and weighting method for robust joint optimization of both consistency and segmentation performance.

### 3.2.1 Consistency as a Surrogate for Generalization

As discussed in Chapter 2, distinguishing between generalizable and non-generalizable predictors, and in turn optimizing for generalizability directly, is not feasible when evaluating only in InD-settings. This is because there is no way of knowing whether the features learned through ERM are causally related to the problem, or if they are simply predictive

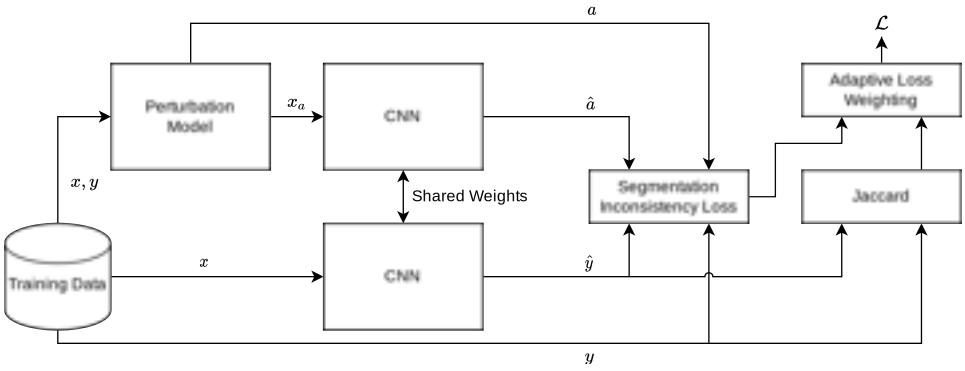


Figure 3.2: Diagram showing Consistency Training. The CNN is given two images, where one is simply an augmented version of the other. It then outputs two segmentations, which in conjunction with the labels for both images is used to compute SIL. The InD IoU for the given batch is then calculated and used to weight this term against a segmentation loss, in our case Jaccard loss.

due to some other correlation that is strong within the bounds of the training distribution. A generalizable evaluation procedure therefore requires some way of determining whether the predictor is leveraging non-causal or causal features.

Determining what features are causally related to the problem is, however, somewhat of an intractable problem. First and foremost, the patterns that neural networks learn and the logic that underpin them are often difficult to identify, and even more difficult to interpret on an intuitive level. Secondly, assuming there was some way of understanding these factors perfectly, establishing causality with any certainty necessitates a higher level of understanding of the problem than is reasonable to expect.

Though establishing what **is causal** is difficult or even impossible, establishing what **is not causal** is not all that complicated. To give a concrete example, consider the problem of classifying images of cows in grassy pastures and camels in deserts. A deep learning model may just as easily learn to associate the "cow" class with grass and the "camel" class with deserts as learning what actually determines the respective animals. Thus, it may predict that a camel standing in a pasture or a savanna is a cow, or equivalently predict that a cow standing in a desert or on a beach is a camel. This is illustrated in Figure 3.3[1].

Associating the cow class with grass and the camel class with sand is obviously non-causal, however, since this pattern would not hold if the model for instance is asked to detect cows on Mars or camels on the Moon. To mitigate this, ones first instinct may be to simply collect data of these

[1]Attributions from top left to bottom right: "Our Camels" by Neil and Kathy Carey, "Cows" by macieklew, "As vacas ditando o caminho" by ground.zero, and "Searching" by Fraser Mummery are licensed under CC BY-SA 2.0.

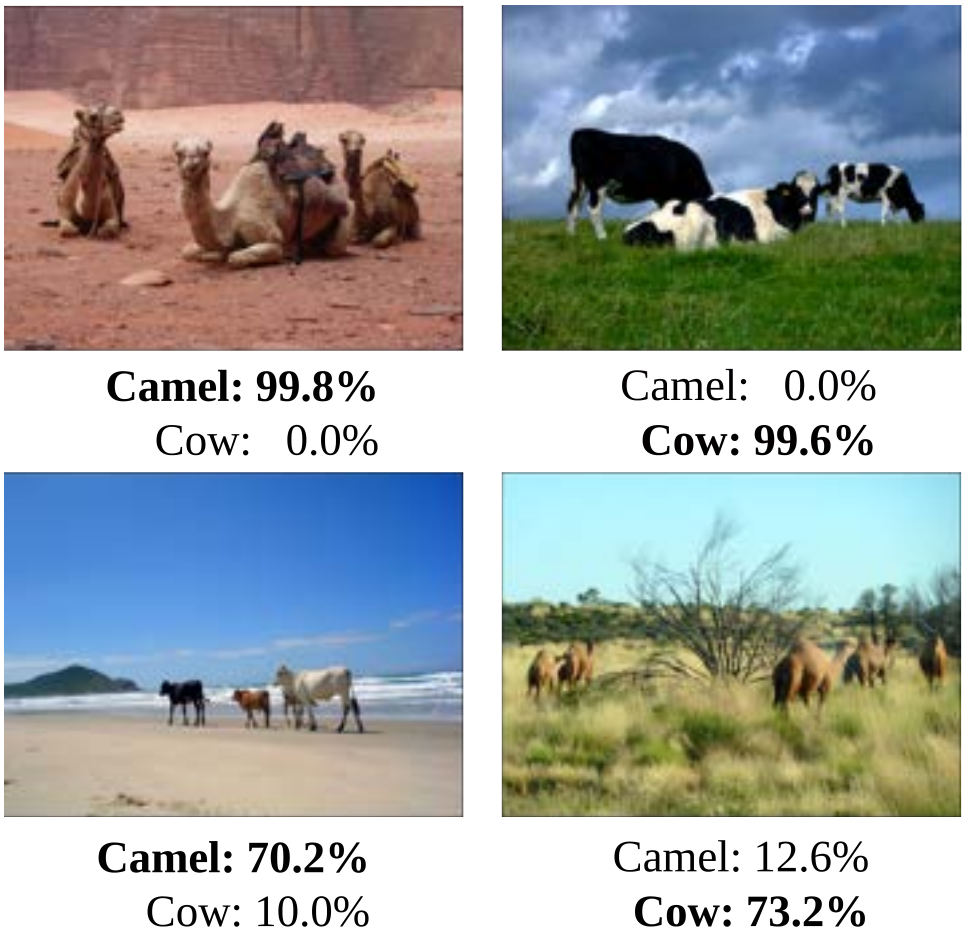


Figure 3.3: A model trained on cows in pastures and camels in deserts may learn to associate the cow class with grass and the camel class with sand, and thus fail to generalize even if performance on an IID test-set is exceptional.

cows and camels in a wide assortment of differing backgrounds, but such careful curation of datasets is not typically feasible, and is at any rate not guaranteed to solve the problem, as another shortcut may easily be found. In the context of polyp segmentation, is for instance not feasible to collect a dataset that is fully representative of all the differing demographics, imaging equipment, endoscopy operator faults, and so on that one may expect in deployment. There is simply too much variation to be fully accounted for. Instead, one has to leverage the data that is actually available and try to squeeze as much utility as possible from it, either by imposing some number of a-priori inductive biases.

Once again going back to the cows and camels example, one may for instance generate multiple instances of the same cow but with varied backgrounds and punish the model for predicting differently depending on the background. This way, the inductive bias that the predictor should

be invariant to backgrounds is imposed.

This, of course, applies to more than just modifying backgrounds: the more of these non-causal changes to the input data are accounted for and modeled, the more spurious correlations are excluded from the search, and the more likely the model is to learn the patterns that are actually causal. These sorts of non-causal changes to the data will from this point on be referred to as *perturbations*. These perturbations can take practically any form, only under the condition that it should not affect the causal structure of the data. If a model is trained such that invariance to all such perturbations is achieved, it must necessarily be leveraging causal features and thus be generalizable. After all, a given set of features can for all intents and purposes be considered causal when they result in performance that holds when subjected to any and all arbitrary perturbations.

Thus, though rewarding causal behaviour is intractable, punishing non-causal behaviour is not. All that is required to do so is to be able to apply perturbations that highlight the non-causal reasoning the model employs, quantify the model's sensitivity to these perturbations, then minimize this quantity through optimization. The resulting model will then in theory have learned invariance to whatever causally irrelevant information that the perturbations define. This property of being invariant to perturbations will be referred to as the *consistency* of the model.

This notion of consistency can in effect be considered a surrogate for generalizability; if a model is consistent to all perturbations, it is invariant to non-causal patterns, and if it is invariant to all non-causal patterns, it necessarily employs causal patterns. Optimizing for consistency can as a result mitigate generalization failure, subject only to the span of the perturbations and how well inconsistent behaviour can be quantified.

This line of reasoning does presuppose that there is some model that can output all possible perturbations one might desire the model to be invariant to. This is of course not the case. As highlighted by the pervasiveness of adversarial attacks and the relative ineffectiveness of adversarial defenses, the perturbations that break DNNs are not necessarily intuitive, and are often difficult to analyze in a manner that is conducive to the task of engineering invariances. Nevertheless, much stands to be gained if the model learns to be invariant even to a fairly limited space of perturbations. Though generalizability is by no means guaranteed in this case, the odds of learning generalizable features are nevertheless improved simply because imposing invariance to a set perturbations limits the types of patterns that a given model can learn. If for instance a white-light endoscopic image is perturbed such that it mimics a narrow-band image, and the model learns to be invariant to this perturbation, predictors that leverage white-light or narrow-band dependent features will no longer be returned from the training procedure.

This approach, then, requires two components: a perturbation model that induces distributional shifts, and a loss function that can describe inconsistent behaviour subject too these distributional shifts. One can then in turn optimize for consistency through gradient decent. The implementation of these two components will be covered in the following sections.

### 3.2.2 Implementing a Perturbation Model

So far, it has been assumed that a perturbation model was given beforehand. This is of course not the case, and any such model needs to be designed with respect to the domain in question. Rotational invariance makes sense for endoscopic images, for instance, but not for classification of hand-written numbers. Thus, in order to engineer such a model, it is first necessary to establish what invariances are desired for the given task. In the case of polyp-segmentation, it is clear that it is necessary to account for variability in for instance lighting, polyp-size, polyp-shape, polyp-location, camera-quality, color-shifts, blurs, optical distortions, and affine transformations. Thus, a model is required that can (more or less) parameterize this variability. Broadly speaking, these transformations can be categorized as follows:

- Pixel-wise variability, which affect only the image, i.e color-shifts, brightness shifts, contrast-shifts, blurs etc. Practically, this corresponds to changes in lighting conditions, camera motion, dye applications, etc.
- Geometric variability, which affects both the image and the segmentation mask, for instance affine transforms and other spatial distortions. Practically, this corresponds to endoscope orientation, optical distortion in the camera, zooming, etc.
- Distributional variability, which affects both the image and the segmentation mask depending on a learned model of the distribution. Practically, this corresponds to the size, shape and location of the polyps

Pixel-wise variability and geometric variability can be modelled fairly trivially through the use of the same transformations typically used in conventional data-augmentation. Distributional variability, however, is somewhat more difficult, and requires a model that can sufficiently represent some characteristic of the distribution. This can for instance be achieved via and cross-dataset style-transfer [75, 79], but this of course necessitates multiple datasets. Given only one dataset, a different method must be used. For a classification task, this could for instance be DeepAugment [34] or a similar technique. DeepAugment, however, cannot

account for the changes in the segmentation mask that should be induced by the augmentations it generates. Consequently, some other generative model wherein the changes in the segmentation mask can be accounted for is required. To this end, a GAN-inpainter can be used.

### GAN-based Polyp Inpainting

As mentioned in Chapter 2, the use of GANs and other distributional modelling in the context of generalization is typically restricted to image-to-image translation, and typically involves transforming an image drawn from one distribution such that it is IID with a second distribution. This, though interesting and no doubt useful assuming several such datasets are available, has limited practical use. It is not necessarily always the case that there exists multiple datasets depicting identical problems, and merely translating between modalities does not as mentioned in Section 2.4 ensure generalizability.

A better approach is to try to model the training set distribution directly, then perturb the data in accordance with this model. For segmentation problems, this can be achieved through training a model to fill some predefined region with pixels that correspond to whatever segmentation target the model is meant to learn, in this case polyps, then perturb a given sample by for instance increasing the polyps' size or adding extra polyps.

To this end, a simple GAN-inpainter was trained. The Generator $G(\cdot)$ and Discriminator $D(\cdot)$ were both implemented with the DeepLabV3+ architecture, and trained using the following loss formulation, where $L_d$ and $L_g$ corresponds to the discriminator and generator loss respectively, and $x$ and $y$ corresponds to masked selections of the input image and output image respectively, where the mask is given by the segmentation label.

$$L_g = 0.001BCE(D(x), y = 1) + 0.999L1(G(x), x) \tag{3.1}$$

$$L_d = \frac{1}{2}\left[BCE(D(G(x), y = 1) + BCE(D(G(x), y = 0))\right] \tag{3.2}$$

In other words, the generator is given an image where the polyp has been masked out, and then learns to fill in the missing area. The resulting region that the inpainter fills in is then compared to the region defined by the polyp as given by the original unmasked image along with the segmentation mask, and the loss is calculated as above.

The inpainter was trained according to the aforementioned loss function using the Adam optimizer and a cosine annealing scheduler with warm restarts. The hyperparameters are shown in Table 3.1

Though the inpainter is trained using masks taken from the segmentation

| Hyperparameter | Value |
|---|---|
| batch_size | 8 |
| learning rate | 0.0001 |
| epochs | 3000 |
| Scheduler $T_0$ | 100 |
| Scheduler $T_{mult}$ | 2 |

Table 3.1: Hyperparameters for GAN-inpainter training

labels, inference must be done by generating a random region that is somewhat polyp-like. This was done by successively and randomly selecting points within a unit square that are a given minimum distance apart from every other point. These points were then sorted according to their order when counting counter-clockwise from the centroid, and splines generated between between every pair of these sorted points. The region defined by this contour was then used as the inpainting target. Figure 3.4 shows some examples of inpainted polyps.

Though this implementation is by no means state-of-the-art, it should nevertheless be sufficient for the purpose of augmentation, considering the principal differences between generated and real polyp images are finer textural details and colour balancing, which are affected by the other augmentations anyway.

#### Geometric and pixel-wise transformations

The data was augmented using the *albumentations* [17] library for python, which defines a large number of transformations for use in deep learning. To establish which of these augmentations are suitable, one first needs to establish which invariances the model(s) in question should exhibit. Table 3.2 below provides descriptions of the invariances required in the model, the albumentation function that corresponds to the required transform, and the hyperparameters used.

The parameters for the respective functions were selected as follows: one transformation was considered at a time, then parameter value(s) that kept the polyp fairly visible but still sufficiently altered were identified. The augmentations then sample between a range given by this maximum to determine the severity for each transformation. The probability of each transformation was set to 1, such that all transformations given in Table 3.2 were always applied, albeit with severity being randomly selected from between zero and the maximum as previously determined. Thus, though all the transformations were always applied, some may have limited effect if the sampled severity was close to zero. Augmentation examples without the inpainter are shown in Figure 3.5.

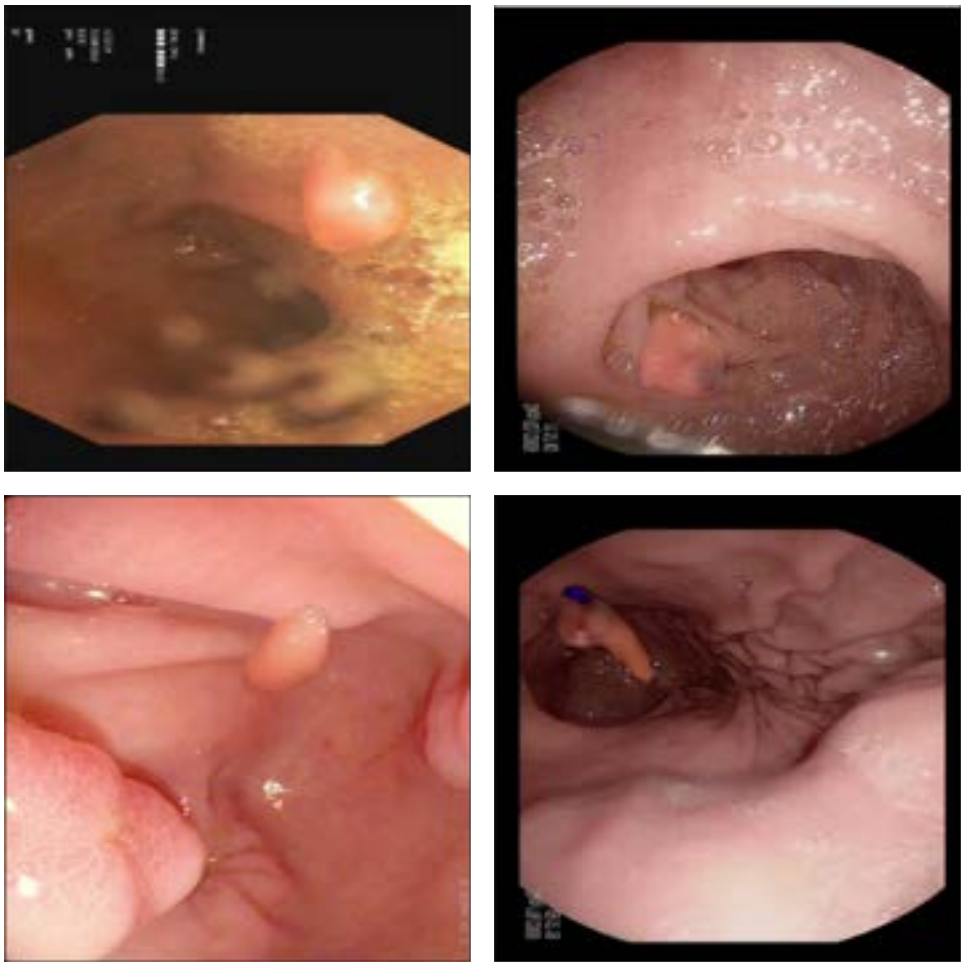

Figure 3.4: Example outputs from GAN-inpainter, given unseen inputs taken from an unlabeled dataset. Besides certain colour artifacts, few textural details, and occasional odd lighting, the generated polyps are moderately convincing.

It should also be noted that this set of augmentations is by no means complete, both in the sense that it accounts for all variability that one might expect in practice, and in the sense that a larger number of augmentations could likely be used. Moreover, the above augmentations are not necessarily optimal, and the selected parameters are not likely to result in the best possible generalizability. In an engineering setting, the choice of augmentations should be tuned and prototyped, but for the purpose of this thesis the relatively simple augmentation strategy as outlined above is sufficient.

| Invariance | Albumentation Function |
|---|---|
| Perspective | Flip()<br>RandomRotate90() |
| Image quality | GaussNoise(max=0.01)<br>ImageCompression(max=100,<br>min=10) |
| Camera models | OpticalDistortion(distort_limit=10) |
| Lighting conditions | ColorJitter(brightenss=0.2,<br>hue=0.2, contrast=0.2,<br>saturation=0.2) |

Table 3.2: Overview of albumentation augmentations functions used in this thesis, with hyperparameters.

Figure 3.5: Sample Augmentations without inpainter.

### 3.2.3 Quantifying Segmentation Consistency

In Section 3.2.1, consistency was defined as the property of exhibiting invariance to perturbations. In the context of segmentation, this corresponds to the ability of the model to output a corresponding segmentation mask when the input data is subjected to some perturbation, such as the augmentations defined in Section 3.2.2.

One simple approach to express this numerically would be to count the number of pixels that do not change in the predicted segmentations when the input is perturbed, and then normalize this with respect to the total number of pixels predicted in both the perturbed and unperturbed images. This, in effect, is equivalent to calculating the IoU across the perturbed and unperturbed segmentations. However, the ground truth may change as a result of the perturbation - if the image is rotated, for example, the

segmentation mask should be rotated accordingly. If an image is globally distorted in some way, the segmentation should exhibit the corresponding distortion. This needs to be taken into account. This can be achieved by discounting the pixels in the predictions that are expected to change from the overall count. This quantity can be expressed as follows:

Let $Y := \{y, \hat{y} := f(x)\}$ be the set consisting of the segmentation labels (masks) and predictions for the unperturbed samples, where $f(\cdot)$ as before denotes the model. Let $\epsilon(\cdot)$ be some perturbation function. Then, let $A := \{a := \epsilon(y), \hat{a} := f(\epsilon(x))\}$ be the set consisting of segmentation predictions and masks when the input is subjected to a perturbation. Segmentation consistency can then be quantified as:

$$\mathcal{C}(y, \hat{y}, a, \hat{a}) = \frac{\sum\{y \cap a \cap \hat{y} \cap \hat{a}\}}{\sum\{y \cup a \cup \hat{y} \cup \hat{a}\}} \tag{3.3}$$

A visualisation of this metric at work is shown in Figure 3.6.

Using this formulation, higher is of course better. For the purpose of developing a loss function, however, it is useful to instead quantify *inconsistency*. This can be expressed in much the same manner, but using the symmetric difference operator, i.e $A \ominus B = A \cup B - A \cap B$:

$$\overline{\mathcal{C}}(y, \hat{y}, a, \hat{a}) = \frac{1}{\sum\{y \cup a \cup \hat{y} \cup \hat{a}\}} \sum\{y \ominus \hat{y} \ominus a \ominus \hat{a}\} \tag{3.4}$$

These formulations are, of course, related by:

$$\mathcal{C}(y, \hat{y}, a, \hat{a}) = 1 - \overline{\mathcal{C}}(y, \hat{y}, a, \hat{a})$$

This notion of inconsistency then corresponds to counting the number of pixels that change after the input is subjected to a perturbation - $\hat{a} \ominus \hat{y}$, but discounting those we expect to change, $a \ominus y$. This is also shown in Figure 3.6 and Figure 3.7.

It is worth noting that consistency is maximized - and thus inconsistency minimized - not only if the predictions are both correct and consistent with one another, but also if the predictions are both incorrect, as long as whatever change that occurs corresponds to the expected change. This is illustrated in Figure 3.7.

Moreover, note that this metric does not presuppose what transformation has occurred. In Figure 3.7, for instance, the change induced by the perturbation may correspond to simply moving the polyp in the image (and replacing the empty space with a believable background), or it may correspond to a rotation by 90 degrees. How this should be analyzed with respect to consistency is up to interpretation - one can argue that a rotation should rotate the incorrect predictions as well, or one can argue

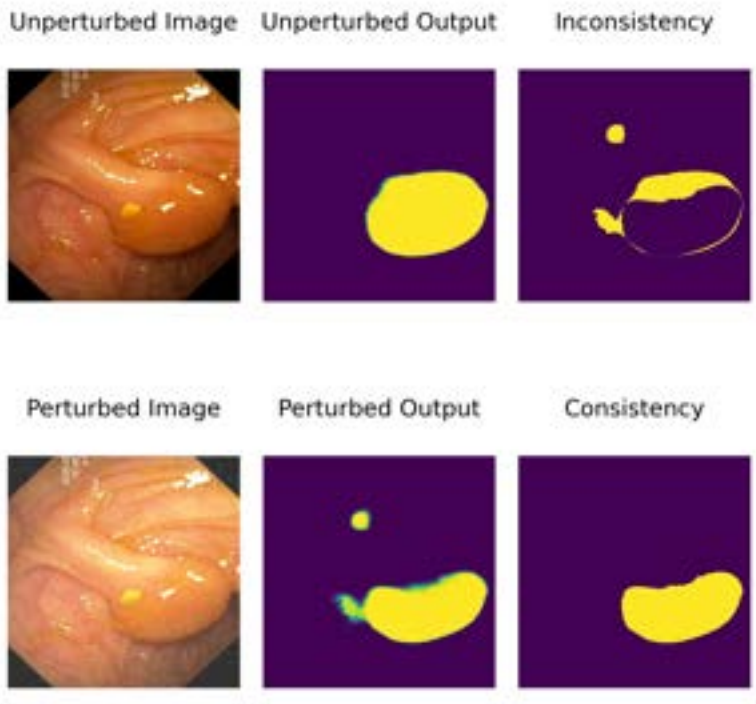


Figure 3.6: Examples of consistency and inconsistency calculation when subjected to a non-label-altering perturbation, in this case additive noise. The consistency for this sample (when thresholded) is 0.68 and inconsistency 0.32, meaning that 64% of the pixels constitute consistent predictions across the two inputs.

that it should only rotate the correct component of the prediction. For simplicity, Consistency Training is based on the latter interpretation. This will be discussed in further detail in Section 6.3.1.

### 3.2.4 Segmentation Inconsistency Loss

Inconsistency as expressed in Equation (3.4) is not differentiable, and thus it cannot in its current state be used as a part of a loss function. Thus, a smooth extension of this metric is needed. This can be achieved in much the same way as how the Jaccard loss can be derived from the IoU - i.e by using differentiable versions of the set functions. We can extend the definition of the symmetric difference to $\Theta(A, B) = A(1 -$

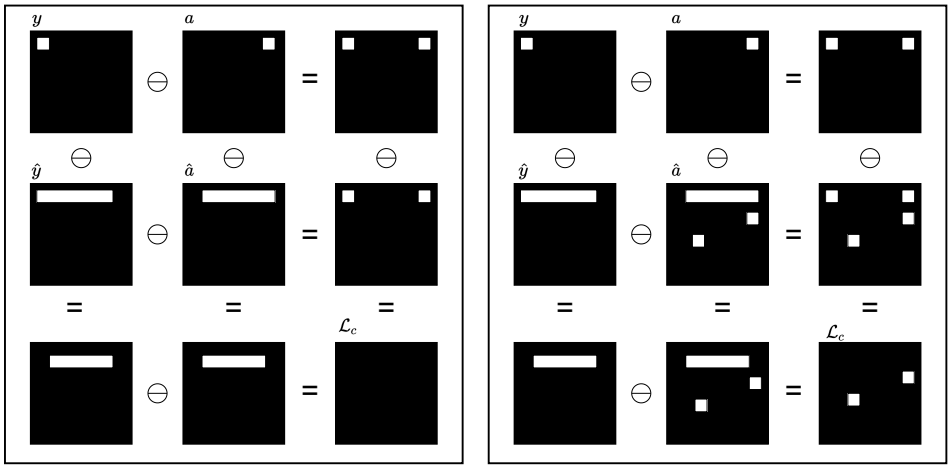


Figure 3.7: Visualization of consistency calculation when subjected to a label-altering perturbation, where white is a positive prediction. Note that $\overline{\mathcal{C}}$ is zero regardless of prediction correctness so long as it changes in the expected manner. Note also that the symmetric difference operators are associative. Left shows an instance of consistent and partially incorrect predictions, and right shows an instance of inconsistent and partially correct predictions.

$B) + B(1 - A)$. This, naturally, is equivalent to the standard symmetric difference if the values of A and B are binary. Similarly, the union operator can be extended as $\bigcup(A, B) = A + B - AB$, and the intersection operator as $\bigcap(A, B) = AB$. Like their binary equivalents, these operators maintain their associative and commutative properties. One can optimize for consistency by replacing the operators in Equation (3.4) with these functions, which in turn can be used as a loss function:

$$L_{SIL}(y, \hat{y}, a, \hat{a}) = \sum \frac{\Theta(y, \hat{y}, a, \hat{a})}{\bigcup(y, \hat{y}, a, \hat{a})} \tag{3.5}$$

This loss function will from this point be referred to as the SIL.

### 3.2.5 Adaptive Loss Weighting

Using SIL as a loss function on its own is not really useful since it only expresses inconsistency, and is to a large extent agnostic to whatever object it is trying to segment. To illustrate, consider a model that predicts that every pixel is positive regardless of the content of the image, and that the augmentation strategy does not make use of augmentations that affect the labels. In this case, the consistency term will always be zero. For example, if the augmentation being performed is simply additive noise, the inconsistency term is equally well minimized if the model learns to

predict that every pixel is positive as it would be if the model learned to be actually be robust to additive noise. Consequently, it has to be combined with a segmentation loss, for instance Jaccard loss. A simple way to do this would be to simply add them together and normalize, i.e:

$$L(Y, A) = \frac{1}{2}\left[L_{seg}((y, \hat{y}) + L_{SIL}(y, \hat{y}, a, \hat{a})\right]$$

Preliminary experiments showed that this, however, exhibited some degree of instability. The model would readily get stuck in local minima where its predictions were indeed consistent, but also consistently predicting artifacts. Examples of this can be found in Appendix C.

To mitigate this, it is possible to employ a weighting strategy. Instead of simply adding the respective losses together, one may weight the individual components adaptively according to the InD segmentation performance, which in this implementation was quantified as the mean IoU of the batch:

$$\mathcal{L}(y, \hat{y}, a, \hat{a}) = (1 - mIoU_{batch}) \times L_{seg}(y, \hat{y}) + mIoU_{batch} \times L_{SIL}(y, \hat{y}, a, \hat{a}) \tag{3.6}$$

Using this formulation, the model will start off trying to learn features that contribute to generally improved segmentation performance, then as segmentation performance improves start principally focusing on learning to be consistent and thus generalizable. If the model starts veering into areas in the loss-landscape that constitute poor segmentation performance, it will self-correct by weighting the segmentation loss more.

### 3.2.6 Conventional Data Augmentation and Consistency Training

At this point, one may easily make the argument that Consistency Training is merely a somewhat more elaborate form of regular data augmentation. To some extent, this argument is well-founded; data augmentations are after all a form of perturbation, and one may argue that ERM is the mechanism by which consistency across these perturbations is minimized. There are, however, a number of nuances that separate the two methods, as will be discussed below.

With conventional data augmentation, one might assume that the model learns to be invariant to the augmentations as a byproduct of minimizing the empirical risk. By extension, it is assumed that the model will learn features that are equally performant across augmentations. After all, the risk-minimizing configuration is in this case that which exhibits the highest degree of performance when averaged across both augmented and non-augmented images.

This, however, is not necessarily the case. To illustrate, consider a pipeline intended to segment melanomas. As mentioned in Section 2.3.1, the

models in such pipelines are often sensitive to skin-tone. Let us assume that the dataset consists primarily of patients with light complexions, and that data augmentation is used in an attempt to combat any bias as a result of this unbalanced dataset. For simplicity, let us assume that the only augmentation used is transforming the image with colour-jitter with probability $p = 0.5$. In theory, the empirical risk will be best minimized by learning features that do not consider colour and thus skin-tone at all, and instead simply learn to consider the shapes and sizes of the melanomas, the irregularity of which is typically considered a the principal hall-mark of melanomas.

This is unlikely for two reasons: first, it presupposes that the model readily learns these generalizable features in favor of the more predictive but spurious features during gradient decent. Second, it assumes that learning to perform well is equally easy on both the augmented and the un-augmented images. If, for instance, the model at an early stage of training learns to use color features to achieve excellent performance on the non-augmented images, while exhibiting mediocre or even poor performance on the augmented data, it is unlikely that the model will ever exit this extremely broad local minimum in favor of a more shape-biased and generalizable configuration. Moreover, it may be the case that shape-based features are more complex to learn, and thus that the performance on the augmented data is limited to a much lower upper bound. In this case, the risk will be minimized not by learning features invariant to the transform, but by learning features that result in a sufficient equilibrium of performance across the augmented and unaugmented sets. I.e, it will try to learn predictive but brittle features as much as possible to maximize performance on unaugmented data, but under the condition that the performance does not degrade too much on the augmented data. Consistency Training mitigates this by explicitly quantifying the inconsistency of the predictor subject to perturbations, and directly minimizing this quantity. Moreover, due to the weighting method used, consistency is also prioritized starting fairly early in gradient descent - as soon as the mean batch IoU exceeds 0.5.

Thus, though the two methods share similar traits, they are distinct. Consistency Training can however be considered an alternative to conventional data augmentation; in segmentation pipelines wherein data augmentation is used, one can implement Consistency Training instead without significant overhead.

### 3.2.7 Putting it all together

To summarize, Consistency Training is based on the idea that a model necessarily must have learned generalizable features if it has learned invariance to all possible perturbations. This is achieved using a perturbation model $\epsilon(\cdot)$, and a loss function which quantifies the inconsistency of the

model when subjected to this perturbation. This loss term then is then incorporated into the final loss function along with a task-specific loss, and weighted according to the model's performance on this task in order to maintain sufficient stability during training. The overall algorithm training process is shown in Algorithm 1:

**Algorithm 1** Consistency Training

**for** *epochs* **do**
  **for** $(batched)\ x, y \in dataset$ **do**
    $x_a, a = \epsilon(x, y)$
    $\hat{y} \leftarrow f(x)$
    $\hat{a} \leftarrow f(x_a)$
    $L_{sil} \leftarrow \frac{\Theta(\hat{x},\hat{a},x,a)}{\bigcup(\hat{x},\hat{a},x,a)}$
    $k \leftarrow IoU(x, y)$
    $\mathcal{L} = (1-k)\mathcal{L}_{\int}(x, y) + kL_{sil}$
    $f(\cdot) \leftarrow weight\_update(\mathcal{L})$
  **end for**
**end for**

## 3.3 Consistency-trained Ensemble Models

As mentioned in Chapter 2, ensemble-based models have demonstrated high degrees of generalizability [41, 88]. Assembling predictors trained with Consistency Training into an ensemble is as a result a simple but effective means by which generalizability can be further increased.

This can be achieved by leveraging multiple identically trained models, such as the dual-decoder DeepLabV3+ - or indeed any model, as will be demonstrated in Chapter 4. As with conventional ensemble models, these predictors can then be used to generate a unique segmentation probability map for each model. This can then be combined into a heatmap, which in turn can be used to facilitate prediction through the use of any number of consensus methods. In this thesis, the consensus method used was a simple majority-vote, i.e. by thresholding the probability heatmap such that all pixels with probabilities above 0.5 were considered as positive predictions. This is illustrated in Figure 3.8.

As mentioned in Chapter 2, ensemble models can be considered a form of Bayesian marginalization. As a result, the model is less likely to be affected by underspecification by virtue of the fact that whatever variability in the space of features that a predictor can learn is to some extent accounted for.

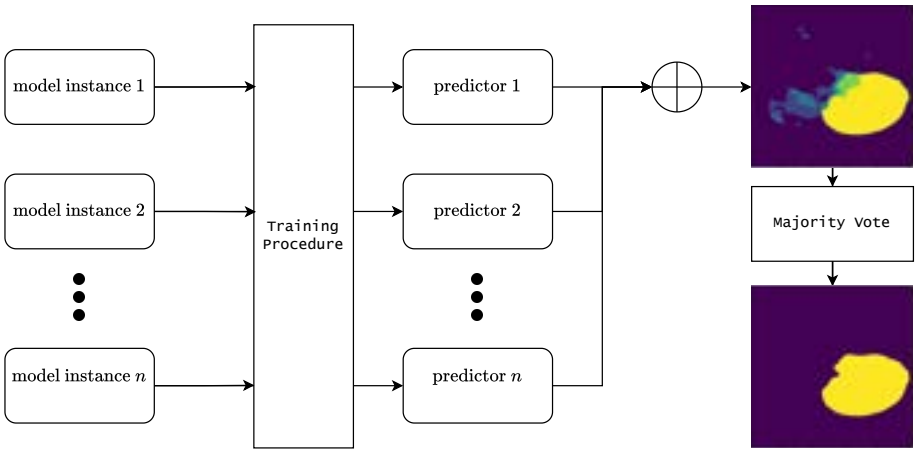


Figure 3.8: Implementation of Ensembles

## 3.4 Summary

This chapter has covered the implementation and theoretical basis for the methods introduced in this thesis. The **dual decoder DeepLabV3+** aims to increase generalization by constraining the models' latent feature space through the use of image reconstruction as an auxiliary task. **Consistency Training** aims to increase generalization by explicitly optimizing for consistent predictions across perturbed and un-perturbed inputs. These perturbations are application-dependent, and are in this thesis implemented as a carefully designed set of augmentations, consisting of conventional image transformations and a generative inpainting model. Finally, a number of **ensemble models** are implemented by combining multiple predictors trained with Consistency Training.

# Chapter 4

# Experiments and Results

This chapter presents the experiments conducted to evaluate the methods presented in Chapter 3, as well as their set up and the experimental methodology used to this end. The results of each experiment are then presented and discussed in brief. Section 4.1 will present the experimental setup used in this thesis, including the choices of metrics, datasets, models, and statistical tests used throughout the experiments. Baseline generalizability metrics for each model are then collected in Section 4.2. The impact of data augmentation on generalization is then tested in Section 4.3, which in turn is used as a basis for the experiments performed in Section 4.4, wherein the best augmentation method was selected for use in Consistency Training. Finally, the impact of ensembles is tested in Section 4.5. All experiments where conducted using Nvidia Tesla-V100 GPUs on the eX3 computing infrastructure offered by Simula Research Laboratory. The experiments were implemented in Python 3.8 using PyTorch 1.8.0. The source code as well as all of the raw data is available on the GitHub repository given in Appendix A.

## 4.1 Experimental Setup

The experiments conducted in this chapter were partially exploratory and partially quantitative in nature. The relative impacts of the methods and baselines on generalization was determined quantitatively where possible through fitting statistical tests depending on the required null-hypothesis and nature of the distribution within the respective groups. Where further analysis was necessary and feasible, these results were then explored in an attempt to relate them to the theory as presented in Section 2.4.

In addition to determining the impact of the methods as outlined in Chapter 3, the effects of the methods that constitute the baselines were also compared to one another. In particular, the effect of the choice of model

architecture, the use of data augmentation, and the use of ensemble models was quantified and related to one another.

An alpha-value of 0.01 was used to ascertain statistical significance throughout this thesis. The p-values for all comparisons performed in this thesis can be found in Appendix B. The statistical tests used in this thesis are as follows:

- Two-sided independent-sample t-tests were used to perform comparisons between approximately normally distributed groups.
- The Mann-Whitney U-test was used to perform comparisons between groups that were not normally distributed, for instance when considering the results across multiple models and/or datasets simultaneously.
- The Spearman's rank correlation test was used to identify correlations, and was selected due to the lack of assumptions of linearity.

The following sections will further detail the experimental setup, including the choices of metrics, datasets, and the choice of models with which the impact of the presented methods were established.

#### Datasets

The only way to sufficiently evaluate the generalizability of a given predictor is to test it directly on OOD data. Though this can to some extent be achieved by carefully designing stress-tests [20], a more straightforward approach is to simply leverage existing OOD datasets. To this end, a number of polyp-segmentation datasets were selected. The names, sizes, resolutions and availability of these datasets is shown in Table 4.1. Samples images and masks from the datasets are shown in in Figure 4.1

| Dataset | Resolution | Size | Availability |
|---|---|---|---|
| Kvasir-SEG [49] | Variable | 1000 | Public |
| Etis-LaribDB [86] | 1255x966 | 196 | Public |
| CVC-ClinicDB [12] | 388x288 | 612 | Public |
| EndoCV2020 [2] | Variable | 127 | By Request |

Table 4.1: Dataset Overview

Kvasir-SEG, the segmentation portion of HyperKvasir [16], was selected as the InD dataset across all experiments due to its size and the diversity of images. This was then split into a training, validation and test-set, which remained constant across all experiments. The remaining datasets were used solely for OOD evaluation.


Figure 4.1: Sample images from the datasets.

All images were resized to 512x512 as preprocessing during all training runs, as some of the models required base-2 dimensionality.

### 4.1.1 Metrics

This subsection will present the metrics used in order to evaluate the performance of the predictors presented in this thesis. As the primary focus is to evaluate generalizability, only two metrics are used, namely the mIoU sample-mean and the Coefficient of Standard Deviation (C.StD) of the mIoU.

### Mean Intersection-over-Union

The most natural way to quantify generalizability is to simply evaluate the predictors on in-distribution and out-of-distribution data and then consider the differences. There are several performance measures that can be used to this end in the context of segmentation, the most natural of which being mIoU or the Dice coefficient, which as discussed in Chapter 2 are equivalent. In this thesis, mIoU was used. To reiterate, IoU is defined as follows: Let $y$ be the segmentation label, and $\hat{y} = f(x)$ be the segmentation prediction given the model $f$ and an input image x. The IoU can then be expressed as:

$$IoU(y, \hat{y}) = \frac{\sum\{y = \hat{y}\}}{\sum\{y = 1\} \cup \{\hat{y} = 1\}\}}$$

Taking the average Mean Intersection over Union (mIoU) over the sample predictors for each dataset should provide an indication of the generalizability of the given pipeline. Though it is of course impossible to account for all distributional shifts that may occur in deployment, high degrees of generalization across multiple datasets should nevertheless translate well to other datasets.

For simplicity, the predictorwise sample mean of the dataset mean IoU will simply be referred to as mIoU throughout the remainder of this thesis.

### Performance Variability

As discussed in Chapter 2, the prevalence of generalization failure is often attributed to the notion of underspecification. Underspeficified pipelines are characterized by the fact that they can return any number of different predictors, which though all exhibiting more or less identical performance in InD settings, learn differing and often conflicting features and thus may differ wildly in OOD settings. To analyze this, the literature tends to consider the performance variability of a set of multiple identically trained predictors [20].

One simple approach to quantify this is to take the standard deviation of the mIoU scores for the given datasets and predictors. This, however, implicitly rewards predictors that perform poorly. To mitigate this, the Coefficient of Standard Deviation (C.StD) can instead be used. C.StD is similar to the standard deviation, but normalized by the mean. This is shown in Equation (4.1), where $n = |\{x_0, x_1, \ldots, x_n\}|$ is the number of samples (in this thesis: mIoU for a given predictor), and $\mu$ is the sample mean (in this thesis: the mean of the mean mIoUs across predictor samples)

$$C.StD = \frac{1}{n\mu}\sqrt{\sum_{i}^{n}(\mu - x_i)^2} \tag{4.1}$$

Though the mean generalizability gap across these predictors is the primary indication of generalizability of the pipeline, this variability is also a salient factor to consider as it serves to quantify the degree to which a given pipeline is underspecified. The more underspecified a pipeline is, the higher the variability of the performance and the higher the C.StD of the mIoUs. For simplicity, this metric will simply be referred to as C.StD in the remainder of the thesis.

### 4.1.2 Models

In order to evaluate the impact of the methods presented in Chapter 3 sufficiently, they need to be tested across a range of different models. This ensures that the effects induced by the methods are not model-dependent, and in addition provides an opportunity to investigate the innate ability of specific models to learn generalizable features. To this end, a number of popular models were selected, intended to serve as a somewhat representative sample of what may be considered as "typical" deep learning pipelines. These models include DeepLabV3+ [18], FPN [62], UNet [76], Tri-Unet [88], and the dual-decoder DeepLabV3+ as introduced in Chapter 3.

The models were implemented in pytorch using the segmentation-models-pytorch (SMP) library [97], using the library's default values. Table 4.2 shows the architecture type and parameter counts of the respective models. The models were all initialized using SMP's built-in pretrained weights,

| Model | Architecture | Parameters |
|---|---|---|
| UNet [76] | Encoder-Decoder | 48 872 738 |
| TriUnet [88] | Stacked Encoder-Decoder | 122 178 709 |
| FPN [62] | Pyramidal | 47 591 762 |
| DeepLabV3+ [18] | Hybrid | 22 437 457 |
| DD-DeepLabV3+ | Single-encoder Dual-decoder | 23 590 756 |

Table 4.2: Experiment Models

trained on ImageNet. Though foregoing pretraining would perhaps highlight the respective models' innate generalization ability to a greater extent, the use of pretrained weights nonetheless constitutes a more realistic context, as most computer vision pipelines, especially those of a medical nature, employ some form of pretraining. As will be discussed in Chapter 5, evaluating the generalizability of different models without pretraining may however be an interesting direction of further study.

## 4.2 Model Architecture

To establish the effect of model architectures alone, ten predictors were trained for each model without augmentation and using regular Jaccard loss, according to the hyperparameters shown in Table 4.3.

| **Pipeline Configuration** | | |
|---|---|---|
| Component | Type | Hyperparameters |
| Dataloader | - | $batch\_size = 8$<br>train/val/test split = 80/10/10 |
| Optimizer | Adam | $lr = 0.00001$ |
| Scheduler | Cosine Annealing w/ Warm Restarts | $T_0 = 50$<br>$T_{mult} = 2$ |
| Evaluation | Loss-based Early Stopping | $epochs = 300$ |

Table 4.3: Hyperparameters for baselines

The mean mIoUs for each dataset are shown in Table 4.4. Though the differences between many pairs of models are statistically significant for several datasets, the magnitude thereof is marginal to the point of being inconsequential for practical purposes, with the exception of TriUnet which exhibited considerably worse generalization. All p-values are shown in Figure B.1.

| Model | Kvasir-SEG | Etis-LaribDB | CVC-ClinicDB | EndoCV2020 |
|---|---|---|---|---|
| DeepLabV3+ | 0.819 | **0.412** | 0.678 | 0.604 |
| DD-DeepLabV3+ | **0.832** | 0.406 | **0.683** | 0.595 |
| Unet | 0.828 | 0.403 | 0.679 | 0.599 |
| TriUnet | 0.822 | 0.305 | 0.633 | 0.581 |
| FPN | 0.823 | 0.404 | 0.678 | **0.605** |

Table 4.4: mIoU scores for each model across datasets. The best models for each dataset are highlighted in bold.

Figure 4.2 shows the models' average change in mIoU across the three OOD datasets with respect to the mIoU of the InD dataset. All models exhibited considerable performance degradation, as expected per the discussion in Section 2.3. Once again, the differences across architectures are fairly marginal with the exception of the TriUnet.

What differences there are across the models, however, can to some extent be understood according to the extent to which the models are underspecified. Figure 4.3 shows the C.StD values for each model and

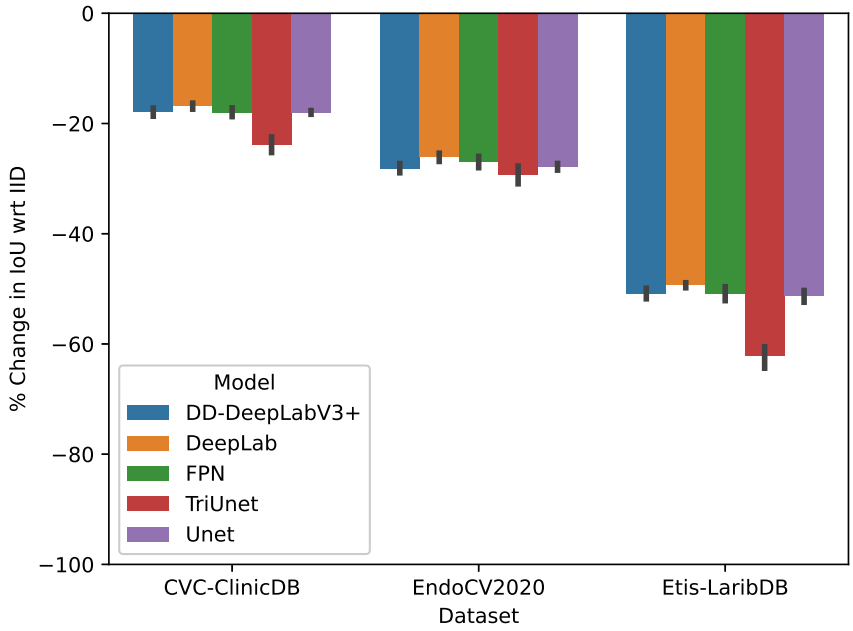


Figure 4.2: Change in OOD mIoU as a percentage of the InD mIoU across models and datasets.

dataset. Evidently, the margins seperating the models in terms of mIoU are similar to the margins sepearating the models in terms of the C.StD. Indeed, as is shown in Figure 4.4 there is a strong negative correlation between the two metrics, suggesting that underspecification plays a considerably more significant role than the model architectures themselves. I.e., the more underspecified the model is, the greater the falls in mIoU are on OOD datasets.

Of particular interest is the relationship between the Unet and the TriUNet, as well as DeepLabV3+ and DD-DeepLabV3+. The differences between these two pairs of models will be discussed in further depth below.

### 4.2.1 Unet vs TriUnet

Consider the differences between the TriUnet and the Unet as shown in Figure 4.3 and Figure 4.2. As the TriUnet consists of three Unets, many analyses would assert that the TriUnet should exhibit equivalent performance or greater, as it affords increased support over the regular Unet. However, the results instead demonstrate that the TriUnet on average performs worse than the regular Unet. This is another piece of evidence that corroborates the notion that underspecification plays a significant role in generalization failure. The TriUnet is fully capable of

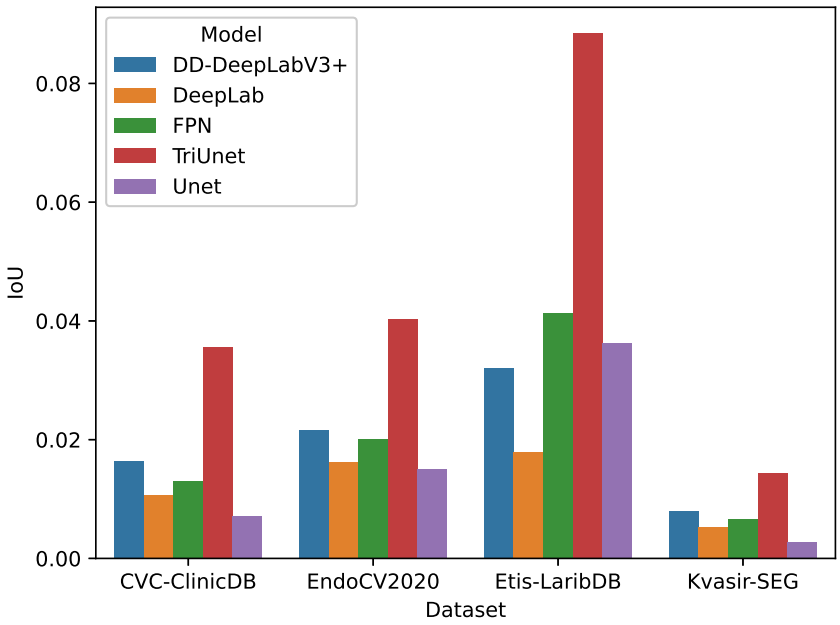


Figure 4.3: C.StD across baseline models and datasets.

learning the same features as the Unet and thus perform similarly, but this does not occur due to underspecification. This is also evidenced by the difference in performance variability between the two models as shown in Figure 4.3.

### 4.2.2 DeepLabV3+ vs DD-DeepLabV3+

DeepLabV3+ and DD-DeepLabV3+ both exhibited more or less comparable performance when considering their OOD IoUs, as shown in Table 4.4. The differneces were not found to be statistically significant. There was however evidently a difference with regards performance variability. As shown in Figure 4.3, the DD-DeepLabV3+ exhibits higher C.StD scores than its single-decoder counterpart, contrary to the hypothesis as presented in Chapter 3.

One possible explanation for these findings is that segmentation encoders may learn somewhat task-agnostic representations of the data by default, and thus that the presence of a reconstruction decoder does not meaningfully affect the segmentation decoder. Following this line of reasoning, the additional decoder may simply increase the degree of underspecification and thus induce performance variability, as it provides additional parameters without meaningfully affecting the features that the model learns during training.

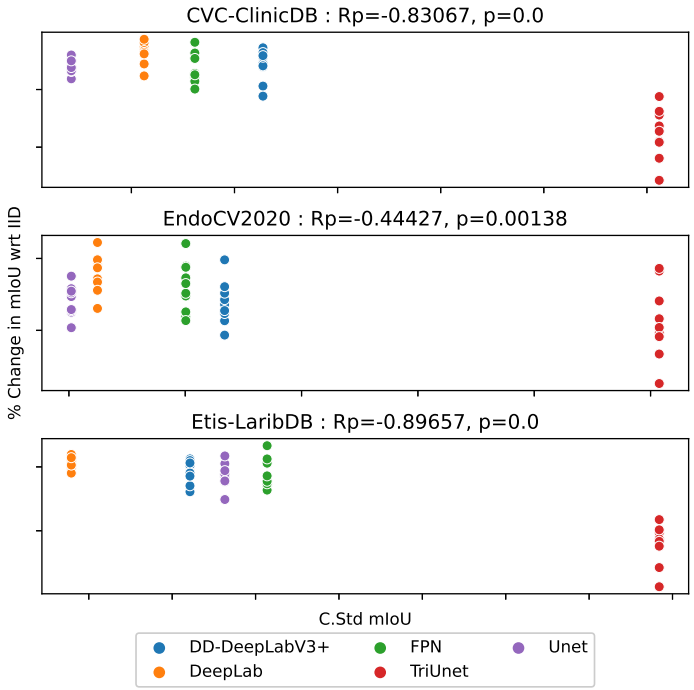


Figure 4.4: Scatter-plot showing the relationship between underspecification as quantified by C.StD and generalization failure as quantified change in mIoU as a percentage of the InD mIoU on models trained without data augmentation.

Though this theory is difficult to verify without analyzing the learned features directly, it is to some extent supported by the fact that the reconstruction seems to be equally good in terms of L1-distance across all four datasets. If the encoder had learned dataset-specific features, this would not be the case. Histograms showing the distribution of L1-scores across the four dataset is shown in Figure 4.6, and reconstruction examples are shown in Figure 4.5.

## 4.3 Augmentation Strategies

The baseline predictors collected in the previous section were then compared to predictors trained using data augmentation. Two augmentation strategies were tested: one with conventional augmentations only, while the other also incorporates the proposed GAN-inpainter, the implementation of which was detailed in Section 3.2.2. The models were trained according to the same hyperparameters as in the previous experiment, shown in Table 4.3. The conventional data augmentation strategy was implemented using albumentations with the same functions and hyperparameters as detailed in Table 3.2. The data was then augmented with a probability of

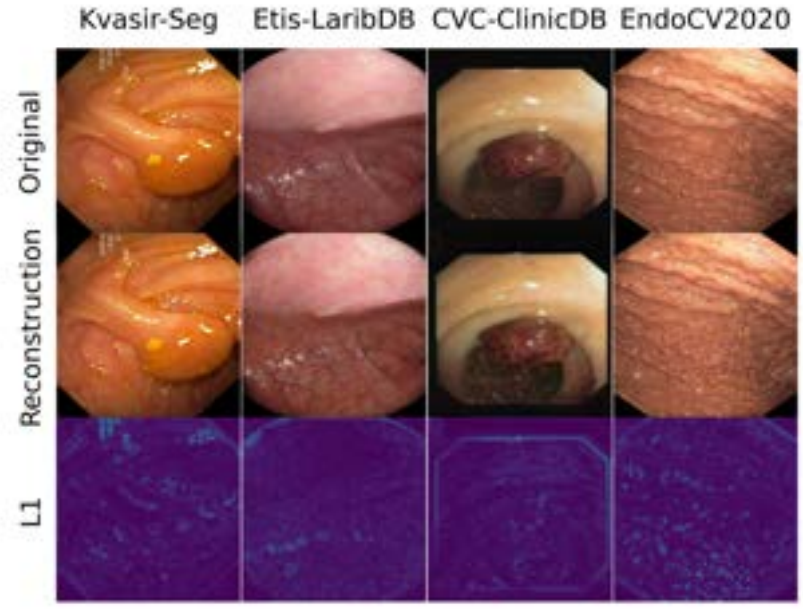


Figure 4.5: Reconstruction Examples across datasets

0.5, in which case the constituent transformations were applied according to the ranges defined by the hyperparameters. The results for each configuration and across models and datasets are shown in Table 4.5.

Both augmentation strategies exhibit an increase in OOD performance when compared to the baseline, i.e no augmentation ($p<0.01$). Averaged across models, the predictors trained using both conventional augmentations and the inpainter perform worse than the predictors trained with the conventional augmentations only on Etis-LaribDB and CVC-ClinicDB ($p<0.01$). There appear to be insignificant differences for the remaining two datasets. The p-values for each dataset can be found in Table B.2. When considering each model individually, the differences are statistically insignificant. The p-values for this can be found in Table B.1.

The relative improvements due to the augmentation strategies as a percentage of the mIoU of the baselines is shown in Figure 4.7.

The difference between the augmentation strategies is best highlighted by the models' performance on Etis-LaribDB, the most difficult of the three OOD datasets, which exhibits increases in mIoU of 7.99% using the inpainter and conventional augmentation and 14.35% using only conventional augmentation. The differences are slightly less pronounced on the CVC-ClinicDB dataset, now with mIoU improvements of 4.55% AND 6.86& respectively, and negligible on the two remaining datasets. One possible reason for this is that the inpainter may have learned InD-

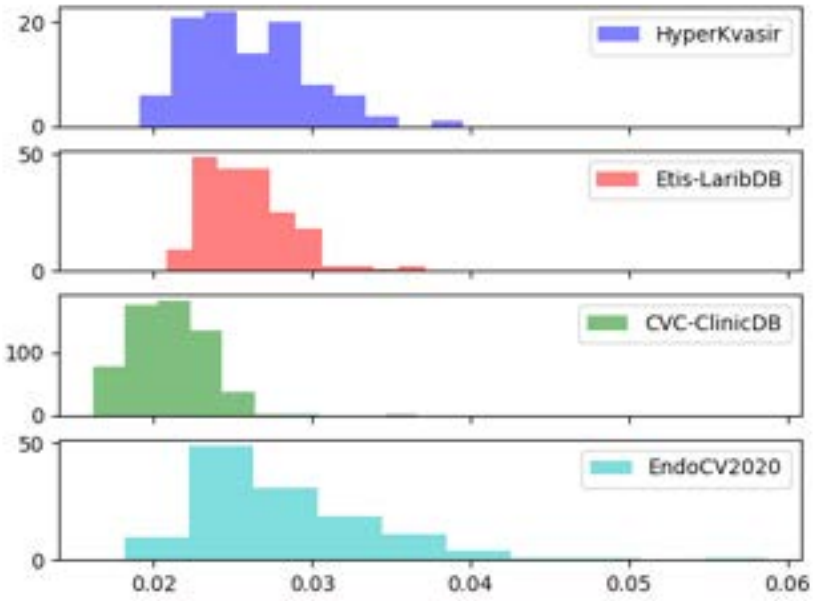


Figure 4.6: Distribution of L1 reconstruction scores across datasets. Though the distributions vary, there is no clear evidence of generalization failure in terms of reconstruction.

specific features, and thus increase the models' bias towards learning these types of features. One can to a minor extent argue that this may explain the limited difference between the two aforementioned augmentation strategies in the InD dataset, i.e KvasirSeg. However, it does not seem to affect the performance on the EndoCV2020 dataset by any significant margin either. One possible explanation for this is that the polyps look similar in both datasets, but verifying this requires further experimentation.

Regardless, it is clear that synthetic augmentation as implemented in this thesis does not benefit generalization. To understand why this is the case, the degree to which the model could identify polyps generated by the inpainter was investigated. To this end, the inpainter was used to add synthetic polyps to the unlabeled portion of HyperKvasir [16]. Though collecting mIoU figures would be uninformative on this dataset, as it also contains unlabeled images of real polyps, a simple visual inspection of the predictions as generated by a model trained with inpainter augmentation, as shown in Figure 4.8, reveals the problem. Evidently, the model fails to recognize inpainted polyps. One possible reason for this is that the risk may have been best minimized by ignoring the inpainted polyps altogether, perhaps because learning features based on the inpainted polyps resulted in higher loss due to the effects thereof on the actual polyps. The inpainted polyps may also be more difficult to segment, as they lack textural details which the model otherwise could leverage, which further complicates

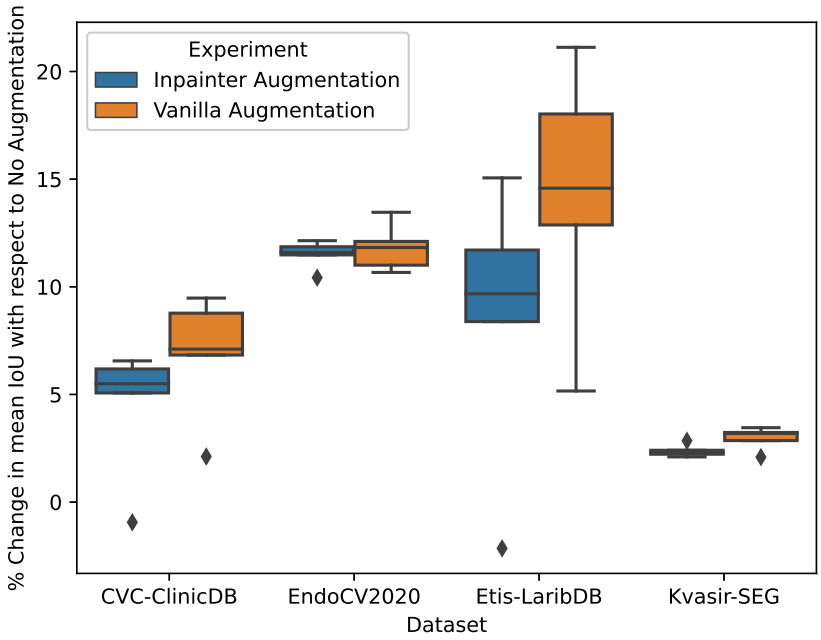


Figure 4.7: Strip Plot of the ensembles' improvements in mIoU per dataset as a percentage of the mIoU of the corresponding model.

matters.

As will be discussed in Chapter 5, however, these results do not however conclusively prove the inefficacy of InD-trained GANs for augmentation as a whole, only that it is unlikely that the implementation in this thesis is particularly useful. The results do, however, demonstrate conclusively that the use of conventional data augmentation contributes a significant increase in generalizability.

## 4.4 Consistency Training

To investigate the impact of Consistency Training, ten predictors were trained therewith. These predictors were then compared to the predictors from the previous experiment trained with conventional augmentation, as Consistency Training in practice is an alternative to data augmentation, and for completeness also predictors trained with no augmentation. As the previous experiment established that conventional augmentations are the most conducive to generalization, it was this strategy that was leveraged as the perturbation model in Consistency Training as well. The mIoUs for this experiment are shown in Table 4.5.

The results show that Consistency Training increases generalization con-

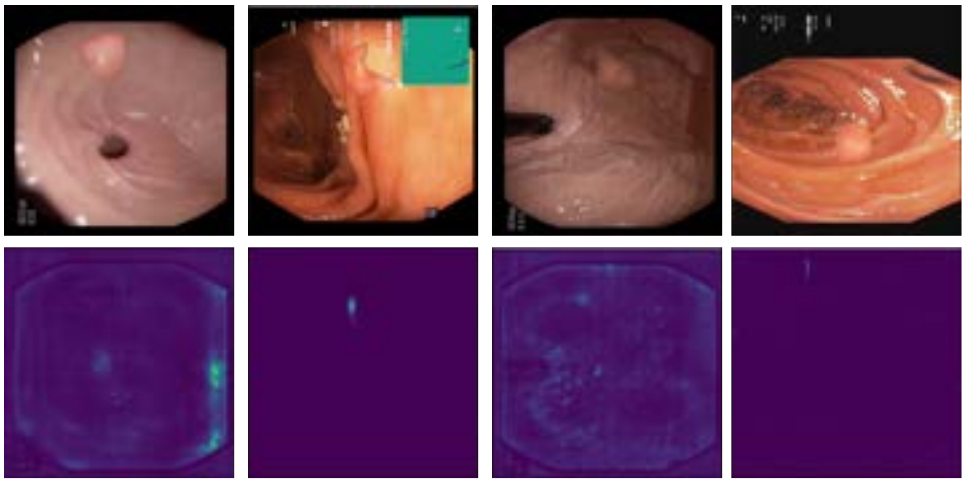

Figure 4.8: Even when trained with inpainting as a part of the augmentation strategy, models do not recognize synthetic polyps.

siderably, outperforming data augmentation by a statistically significant margin on all OOD datasets when comparing across all models. This is shown in Figure 4.9. The p-values can be found in Table B.3.

When analyzing the improvements for the individual models, statistical significance was achieved for all models except the TriUnet on the Etis-LaribDB dataset, for the FPN and Unet on the CVC-ClinicDB dataset, and for the Unet on the EndoCV2020 dataset. The p-values for these comparisons are found in Table B.4.

As discussed in Section 3.2.6, Consistency Training can be interpreted as imposing a more credible set of inductive biases by explicitly optimizing for consistency across augmentations. This is evidenced by considering the performance variability across the configurations, shown in Figure 4.10, which shows that predictors trained with Consistency Training exhibit lower performance variability than conventional data augmentation on two of the three OOD datasets. As the C.StDs are computed from sample standard deviations, there is as discussed inherently some measurement error. It should therefore be noted that the differences between the the C.StDs values cannot be confirmed to statistical significance due to wide confidence intervals of the standard deviations as computed at this sample sizes (N=50) involved in this thesis. This will be discussed further in Chapter 5.

## 4.5 Ensembles

Finally, the impact of combining multiple predictors into an ensemble was investigated. Two types of ensembles were investigated: ensembles

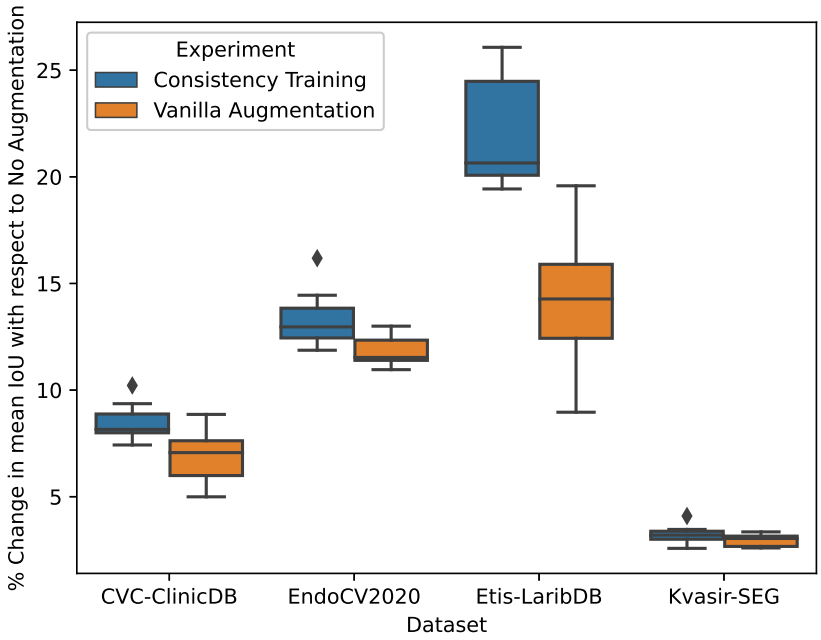


Figure 4.9: Improvements due Consistency Training and Data Augmentation as a percentage the mIoU without augmentation across datasets

consisting of five instances of a single type of model and ensembles consisting of all five models. The generalizability of these ensembles were then compared to one another and with the average performance of their constituent predictors. Such analysis was performed across all of the the training-methods tested in Section 4.4, i.e. no augmentation, conventional data augmentation, and Consistency Training. Finally, the relationship between the improvement due to ensembles and underspecification was explored.

To ascertain the generalizability of the ensembles to statistical significance, ten ensembles of each kind were implemented. For the multi-model ensembles, each of the ten ensembles was built from unique predictors trained in Section 4.4. For the single-model ensembles, five predictors were randomly selected from the ten that were trained in Section 4.4. As will be further discussed in Chapter 5, these ensembles too should have been built from unique predictors, however due to limitations with regards to computational resources this was infeasible.

### 4.5.1 Improvements over Single Models

First, the generalizability of ensemble models was compared to that of the single models. As previously noted, this was performed on pairs

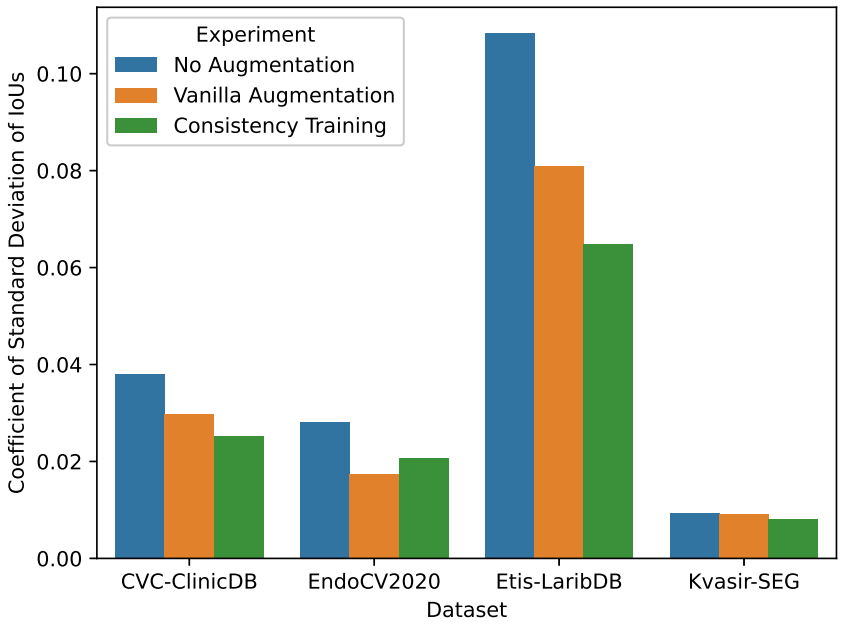


Figure 4.10: Models trained with Consistency Training exhibit lower predictor-wise performance variability than models trained without augmentation or with regular data augmentation

of ensembles and single models across all the three training methods compared in Section 4.4. The mIoU-scores for the ensemble models are shown in Table 4.7. Averaging across models, the ensembles exhibited increased mean mIoUs on all of the OOD datasets when compared to the mIoU of the constituent models as shown in Table 4.6 ($p<0.01$). See Table B.5 for p-values. This corroborates the findings in other works that ensembles contribute to increased generalization [41, 88].

### 4.5.2 Effect of Ensemble Training Methods

The difference in mIoU between the three ensemble training methods was statistically significant on all datasets ($p<0.01$), except CVC-ClinicDB, wherein the difference between the ensembles trained with Consistency Training and the ensembles trained with conventional data augmentation had a p-value of 0.012. The p-values can be found in Figure B.2.

The difference between the relative improvements across the three training methods were statistically insignificant ($p>0.01$), with the average change in IoU as a percentage of the mIoU of the constituent models being 2.026%, 3.081% and 2.351% respectively across all datasets for ensembles trained with no augmentation, conventional augmentation, and Consistency

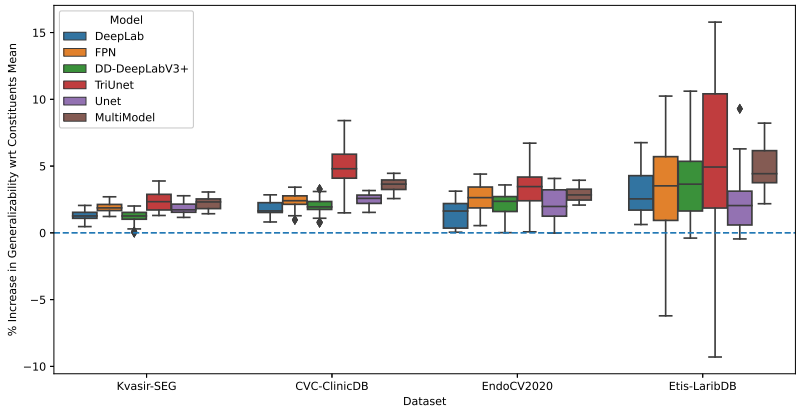


Figure 4.11: Boxplot showing the improvement due to ensembles as a percentage of the mIoU across their constituent models across all three training methods

Training. This is shown in Figure 4.12. The p-values are shown in Figure B.3.

To further analyze the impact of ensembles with respect to single models, one can consider the relative performance improvements between them. To this end, the performance of the ensembles trained with Consistency Training was compared to the mean performance of its constituent predictors across all tested model architectures. The relative improvements as a percentage of the constituent predictor performance are shown in Figure 4.11. The results show that ensembles in the majority of runs increase generalization. However, this is not always the case; perhaps counter-intuitively, the use of ensembles reduce generalization on some datasets in certain runs. This occurred on some of the samples on all ensemble types with the exception of the DeepLabV3+ and multi-model ensembles. This may happen when there are high degrees of disagreement among the constituent predictors, in which case there may not be sufficient consensus to fully segment the polyp. Many implementations of ensembles, therein the implementation used in this thesis, require at least a 50% consensus in order for a given pixel to be classified positively, and thus if this is not achieved, the ensemble may perform worse than any one of the constituent predictors.

### 4.5.3 Ensembles and Underspecification

The tendency of ensembles to increase generalization is as mentioned in Chapter 2 often attributed to the fact that the use of ensembles to some extent mitigate underspecification. Specifically, they constitute a form of Bayesian marginalization, and should thus in theory be able to leverage the

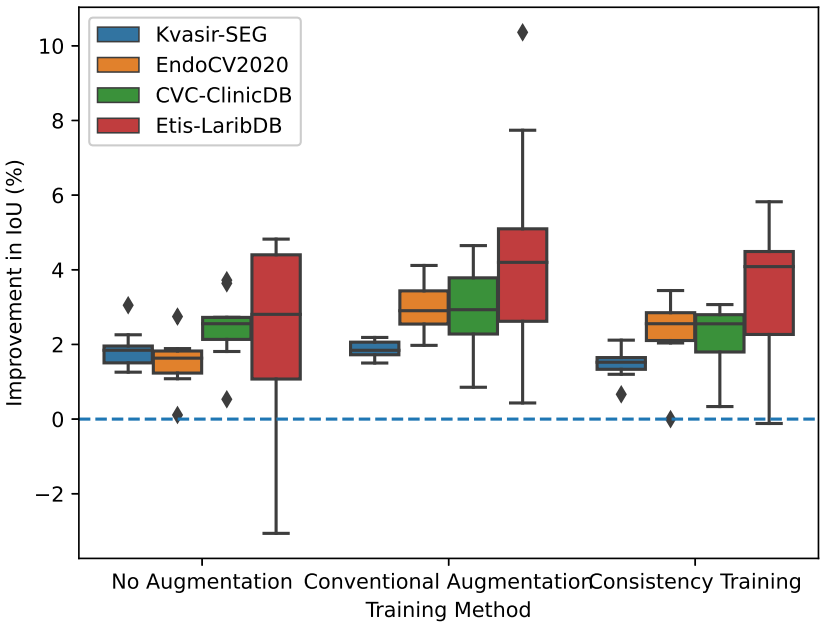


Figure 4.12: Ensemble improvements across training methods and datasets as a percentage of the mIoU of the corresponding model architecture

variability of its constituent predictors in order to mitigate generalization failure to a degree. This assumes that predictions with high consensus are the most generalizable, though as shown in Figure 2.12 this may not be the case.

To further investigate the veracity of this line of reasoning, one can consider the relationship between the improvements to generalization due to the use of ensembles versus the degree to which the pipelines that generate the constituent predictors are underspecified. This is shown in Figure 4.13. The results from a spearman-rho test is shown in the figure title.

There appears to be a positive correlation between the two ($p<0.01$), which corroborates the aforementioned interpretation. It should be noted that the C.StD values are computed based on all ten samples, as it is supposed to represent the degree to which the pipeline itself is underspecified, and not the variability of the constituent predictors for each ensemble instance. One can instead consider the variability in performance of the constituent predictors, which it can be argued is a better representation of the diversity of features learned by the ensembles. This is shown in Figure 4.14. The p-values after a Spearman's $\rho$ test are shown above each subplot.

The C.StD values are in this case based on five predictors - half as many as in Figure 4.11, thus there may be a larger degree of measurement error along the x-axis. Nevertheless, there overall still appears to be a generally

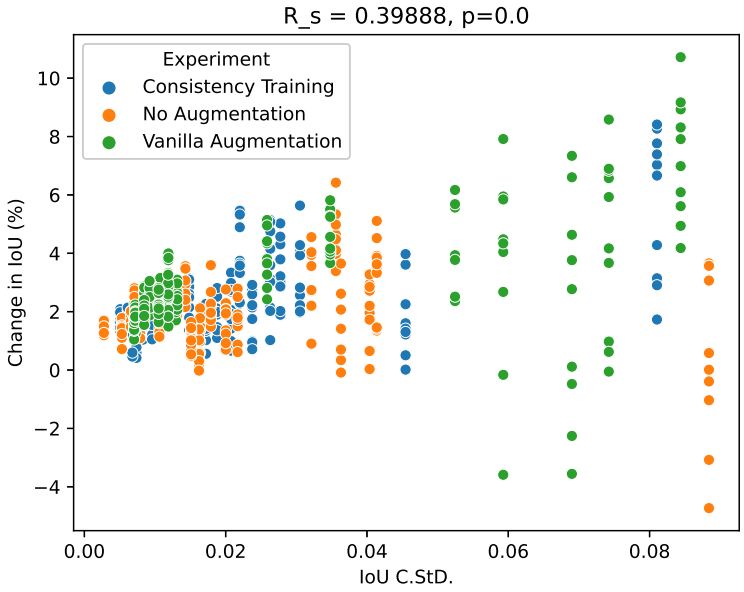


Figure 4.13: Plot showing the relationship between predictor-wise performance variability and improvements in generalization due ensembles trained with Consistency Training. The error-bars show the 99% confidence intervals for the improvement. The more under specified the pipeline is as quantified by performance variability, the greater improvements are made through the use of ensembles.

positive relationship between the ensemble constituent's C.StD of mIoU and the relative improvements in mIoU due to ensembles. When the ensembles are trained either with or without conventional augmentation, this relationship is statistically significant for all datasets. When trained with Consistency Training, it is statistically significant for Etis-LaribDB and CVC-ClinicDB. The weak correlation in the remaining datasets may be attributed to the fact that the models generally perform with low degrees of variability on them, as shown in Figure 4.10. This low variability suggests that the predictors all return fairly similar segmentations, which also explains the comparatively low impact of the ensembles on these datasets.

## 4.6 Summary

This chapter detailed the experiments performed to evaluate the methods presented in Chapter 3 along with the effects of model architecture, augmentation, and ensembles on generalizability. The results can be summarized as follows:

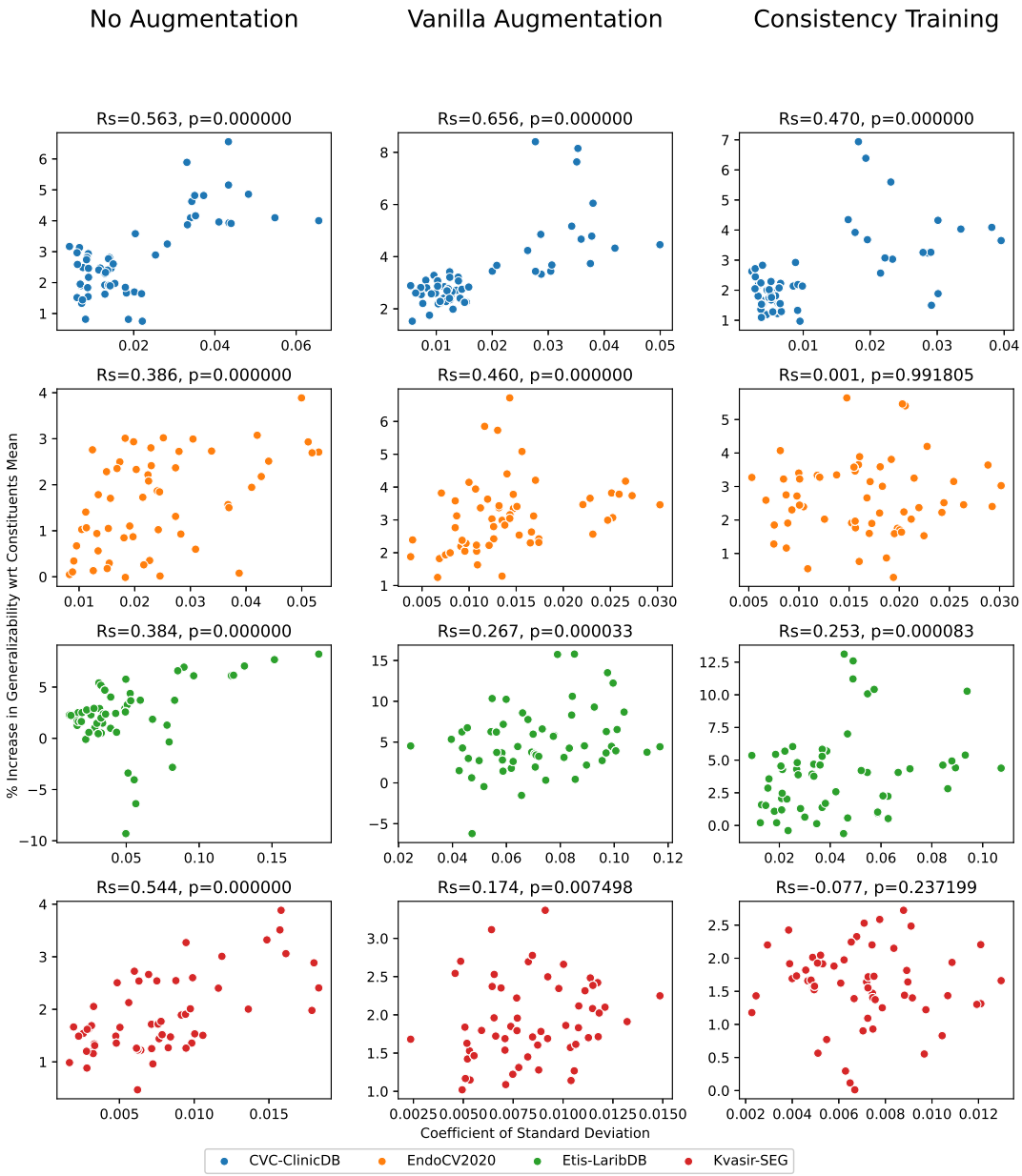


Figure 4.14: Plot showing the correlation between the improvements to mIoU with respect to the mIoU of the constituent predictors, versus the variability in the performance of the constituent predictors. The Spearman Correlation Coefficient and corresponding p-value for each dataset is shown in the title of each subplot.

- Model architecture had limited bearing on increasing generalization, except with regards to the degree to which they are underspecified.
- Multitask learning as implemented using the Dual-decoder DeepLabV3+ had negligible impact, which may be attributed to the encoder learning dataset-agnostic features.

- Data augmentation increased generalization considerably, but the use of the generative inpainter had a negative effect.
- Consistency Training outperformed conventional data augmentation and increased generalization by statistically significant margins.
- Ensembles models increased generalization. The relationship between this increase and underspecification was investigated, and shown to be positively correlated to statistical significance.

These impact and limitations of these findings will be discussed in Chapter 5, as well as limitations of the experimental methodology.

| Model | No Augmentation | Vanilla Augmentation | Inpainter+Vanilla Augmentation |
|---|---|---|---|
| | | **Kvasir-SEG** | |
| DD-DeepLabV3+ | 0.829 | **0.848** | 0.844 |
| DeepLabV3+ | 0.822 | **0.850** | 0.846 |
| FPN | 0.822 | **0.853** | 0.848 |
| TriUnet | 0.817 | 0.841 | **0.842** |
| Unet | 0.828 | **0.851** | 0.846 |
| | | **Etis-LaribDB** | |
| DD-DeepLabV3+ | 0.408 | **0.460** | 0.435 |
| DeepLabV3+ | 0.417 | **0.472** | 0.451 |
| FPN | 0.404 | **0.440** | 0.422 |
| TriUnet | 0.309 | **0.410** | 0.382 |
| Unet | 0.403 | **0.447** | 0.414 |
| | | **CVC-ClinicDB** | |
| DD-DeepLabV3+ | 0.681 | **0.728** | 0.713 |
| DeepLabV3+ | 0.684 | **0.733** | 0.718 |
| FPN | 0.675 | **0.715** | 0.705 |
| TriUnet | 0.623 | **0.684** | 0.659 |
| Unet | 0.679 | **0.717** | 0.703 |
| | | **EndoCV2020** | |
| DD-DeepLabV3+ | 0.596 | **0.668** | **0.668** |
| DeepLabV3+ | 0.608 | **0.676** | 0.670 |
| FPN | 0.600 | **0.662** | 0.661 |
| TriUnet | 0.577 | **0.667** | 0.656 |
| Unet | 0.598 | 0.660 | **0.665** |

Table 4.5: Mean IoUs across augmentation strategies grouped by model and dataset. The best augmentation strategy for each dataset and model are highlighted in bold.

| Model | No Augmentation | Vanilla Augmentation | Consistency Training |
|---|---|---|---|
| | | **Kvasir-SEG** | |
| DD-DeepLabV3+ | 0.829 | 0.848 | **0.852** |
| DeepLab | 0.822 | 0.850 | **0.852** |
| FPN | 0.822 | **0.853** | 0.852 |
| TriUnet | 0.817 | 0.841 | **0.845** |
| Unet | 0.828 | **0.851** | **0.851** |
| | | **Etis-LaribDB** | |
| DD-DeepLabV3+ | 0.408 | 0.460 | **0.482** |
| DeepLab | 0.417 | 0.472 | **0.505*** |
| FPN | 0.404 | 0.440 | **0.475*** |
| TriUnet | 0.309 | 0.410 | **0.434** |
| Unet | 0.403 | 0.447 | **0.481*** |
| | | **CVC-ClinicDB** | |
| DD-DeepLabV3+ | 0.681 | 0.728 | **0.736** |
| DeepLabV3+ | 0.684 | 0.733 | **0.740** |
| FPN | 0.675 | 0.715 | **0.727*** |
| TriUnet | 0.623 | 0.684 | **0.696** |
| Unet | 0.679 | 0.717 | **0.730*** |
| | | **EndoCV2020** | |
| DD-DeepLabV3+ | 0.596 | **0.668** | **0.668** |
| DeepLab | 0.608 | **0.676** | **0.676** |
| FPN | 0.600 | 0.662 | **0.673** |
| TriUnet | 0.577 | 0.667 | **0.684** |
| Unet | 0.598 | 0.660 | **0.676*** |

Table 4.6: Mean IoUs for training methods, precision truncated to 99% confidence. Consistency Training entries with greater performance than conventional augmentation by for the given model and dataset are highlighted in bold. If statistically significant after a two-tailed independent sample t-test, they are also marked with a "*"

| Ensemble | CVC-ClinicDB | EndoCV 2020 | Etis-LaribDB | Kvasir-Seg |
|---|---|---|---|---|
| **Consistency-Trained** | | | | |
| DD-DeepLabV3+ | 0.748 | 0.684 | 0.492 | 0.863 |
| DeepLabV3+ | **0.751** | 0.683 | **0.523** | 0.859 |
| FPN | 0.739 | 0.685 | 0.478 | 0.868 |
| Unet | 0.744 | 0.694 | 0.494 | 0.868 |
| TriUnet | 0.723 | **0.715** | 0.468 | 0.859 |
| MultiModel | 0.747 | 0.693 | 0.484 | 0.867 |
| **Conventional Augmentation** | | | | |
| DD-DeepLabV3+ | 0.746 | 0.685 | 0.480 | 0.861 |
| DeepLab | 0.750 | 0.692 | 0.492 | 0.862 |
| FPN | 0.732 | 0.684 | 0.457 | **0.869** |
| TriUnet | 0.715 | 0.692 | 0.440 | 0.860 |
| Unet | 0.735 | 0.677 | 0.457 | 0.867 |
| MultiModel | 0.740 | 0.687 | 0.462 | 0.867 |
| **No Augmentation** | | | | |
| DD-DeepLabV3+ | 0.690 | 0.605 | 0.419 | 0.840 |
| DeepLab | 0.695 | 0.611 | 0.426 | 0.833 |
| FPN | 0.690 | 0.610 | 0.416 | 0.837 |
| TriUnet | 0.651 | 0.588 | 0.311 | 0.841 |
| Unet | 0.696 | 0.604 | 0.409 | 0.841 |
| MultiModel | 0.694 | 0.612 | 0.414 | 0.846 |

Table 4.7: IoUs across ensemble models, datasets, and training methods. Best ensembles for each dataset are highlighted in bold.

# Chapter 5

# Analysis and Discussion

This chapter will summarize the key findings presented in Chapter 4 and analyze them with respect to the theory as outlined in Chapter 2. The chapter will be organized according to the experiments performed, with each section discussing the results, impact, and limitations of the corresponding experiment. The chapter will start with the results from the individual experiments, including the impact of model architectures on generalization as presented in Section 4.2, the impact of augmentation as presented in Section 4.3, inconsistency Training as presented in Section 4.4, and finally ensembles as presented in Section 4.5. Afterwards, the generalizability of the best performing configuration tested in this thesis will be discussed and considered from a practical perspective.

## 5.1 Model Architectures and Generalizability

The experiments performed in Section 4.2 show that every model exhibited comparable levels of generalization failure, with the exception of TriUnet which seemed to struggle more than any of the other models. On Etis-LaribDB, which evidently proved to be the most difficult dataset, with the models on average exhibiting reductions in generalizability of 52.72%, with the TriUnet ranging upwards of 68.76%. The degree of generalization failure was slightly less pronounced on the two other datasets, with CVC-ClinicDB exhibiting average reductions of 18.78% and EndoCV2020 27.70%.

The models exhibited comparable performance in InD settings, spanning between IoUs of 0.817 and 0.829, which for practical purposes can be considered negligible.

### 5.1.1 Impact

These results highlight that researching the development of more and more complicated task-agnostic models is a comparatively fruitless affair. The difference between DeepLabV3+ and Unet - which are separated by two years of research - are practically inconsequential. Admittedly, the differences are more pronounced when the models are trained according to the more sophisticated training regiments used in the remaining experiments, but it nonetheless does not appear as if it is advancements in model architectures that is likely to result in increased generalization, but rather improvements to the pipeline with which they are trained. There are some limitations to this claim, however, which will be discussed in the context of limitations to the experimental methodology in Section 5.6.

### 5.1.2 Limitations

A largely neglected but nevertheless impactful aspect of the deep learning pipelines studied in this thesis is the use of pretraining. Across all the experiments performed in this thesis, every predictor was pretrained on Imagenet, with the pretrained weights being included in the segmentation-models-pytorch library [97]. Without pretraining, preliminary work showed that the models selected in this thesis exhibited mIoUs of at best around 0.6 at best even on IID, with even more significant performance gaps on OOD data. Non-pretrained networks are for this reason rarely used. However, this pretraining may play a key role in certain aspects of the behaviour observed in this thesis. In particular, pretraining may be the principle contributing factor behind the apparent ineffectiveness of multitask learning. An Imagenet pretrained encoder would, after all, perform the best when practically performing image compression.

## 5.2 Data Augmentation and Generalizability

The experiment in Section 4.3 demonstrated the efficacy of data augmentation as a means of increasing generalization, with an mIoU improvement of 9.00% compared to the pipeline without data augmentation when averaged across models and datasets. This, as mentioned in Section 2.4.7, can be attributed to the fact that a wider diversity of data limits the space of viable features that a model can learn.

The inpainter as implemented in this thesis was, however, proven to be ineffective. The segmentation models appeared to learn to neglect the synthetic polyps. It was hypothesized that this was due to the polyps lacking sufficient similarity to real polyps, which only affected the training procedure such that generalization was harmed.

### 5.2.1 Impact

The extent to which augmentation improved generalization in this thesis was considerable, especially in comparison to some of the other tested methods. In particular, the effects of model architectures and ensembles were both comparatively small. The use of ensembles, for instance, the use of which was the basis of several of the papers submitted to EndoCV2021, increased generalization by at most 10.36% and on average 2.48%, whereas the use of data augmentation led to increases of at most 19.57% and on average 9.00% when compared to no augmentation.

Thus, the margins by which the use of augmentation affects generalization are far greater than the margins by which ensembles affect generalization. As an ensemble-based model was the winning submission to EndoCV2021 [88], it may also be the case that it also has a greater impact than many of the other methods presented in therein affect generalization. As EndoCV2021 did not account for any differences in the participants choice of augmentation strategy when comparing submissions, one can raise questions as to the veracity of its findings. It may, for instance, be the case that certain submissions exhibited high degrees of generalization not strictly because of the impact of their proposed methods, but rather due to their choice of augmentations. This is of course not a certainty, and does as such warrant further research for instance in the form of a meta-analysis.

In the context of the inpainter, the primary takeaway is that further experimentation is needed. Though the inpainter as implemented in this thesis harmed generalization, it may be the case that a more sophisticated implementation, perhaps with pre- and post-processing, could still facilitate increased generalization. Moreover, evaluation of such models needs to be performed with care, as brief qualitative appraisals of performance may as evidenced in this thesis be misleading especially when they do not originate from from a domain expert.

### 5.2.2 Limitations

Ignoring the inpainter and its flaws as outlined above, only one implementation of data augmentation was used throughout this thesis. The constituent transformations and the values of the hyperparameters thereof were also selected with limited prototyping or testing. There may as such be augmentation configurations that induce significantly increased generalization. By the same token, the selection of transformations used in this thesis may instead have been lucky and thus over-represent the typical contribution of data augmentation. A robust investigation of data augmentation and its effects would require a larger range of augmentation strategies. The results thereof would, however, only be of relevance to the particular task that is being considered. Polyp segmentation may benefit more from aug-

mentation than image-captioning, for instance.

Additionally, the augmentations in this thesis were applied according to a predetermined probability. A more effective technique may be to augment every sample, but account for the severity through the modulation of the hyperparameters of the constituent transformations. This was not, however, investigated in this thesis, as the probability-based implementation facilitated more apples-to-apples comparison to Consistency Training.

## 5.3 Consistency Training and Generalizability

Consistency training was shown to improve generalizability, outperforming data augmentation by a significant margin on all OOD datasets. This can be attributed to the additional inductive that are imposed.

### 5.3.1 Impact

Though Consistency Training did increase generalization by a considerable amount, the OOD performance is nevertheless insufficient for practical purposes. The best performance on Etis-LaribDB with Consistency Training was after all merely 0.504, as shown in Table 4.6. This kind of performance would be of limited utility in clinical applications.

Consistency Training does, however, constitute a step in the right direction. In contrast to competing methods such as Model-Based Robust Deep Learning [75], Invariant Risk Minimization [5], or multi-domain training [31], Consistency Training only requires a single dataset, and can as a result be used in practically every segmentation pipeline. The implementation thereof is also conceptually simple, and can for practical purposes be considered a more generalizable alternative to data augmentation.

Given further development, Consistency Training may prove a promising candidate as a means of alleviating generalization failure to practically viable extents, especially if leveraged in conjunction with other methods. As established in Chapter 3, the limits are in theory only the efficacy of the quantification of consistency for a given task and the extent to which the augmentation strategy can account for any distributional shifts that may occur. Improvements to either of these aspects are likely to contribute to considerable gains in generalizability.

Developing perturbation models and consistency metrics may also be a great opportunity to incorporate expert input. A clinician could for instance offer insights as to the nature of the perturbations one might expect in practice and thus assist in the development of the perturbation

model.

### 5.3.2 Limitations

During the experiments performed in this thesis, the batch size was set to eight for all training procedures. As Consistency Training relies on generating pairs of data from a given batch, one may argue that keeping the batch size the same may result in a weak comparison. The experiment should as such ideally be repeated across a number of batch sizes, but this was infeasible due to constraints with regards to computational resources.

Consistency Training was also throughout the thesis treated as an alternative to data augmentation. It may, however, be possible to also augment the incoming batch in the dataloader, and use consistency training as a supplementary method. This way, it will also optimize for consistency across differently augmented samples.

## 5.4 Ensembles and Generalizability

The use of ensembles, as shown in Section 4.5, was shown to increase generalization. The improvements were on average comparatively minor, however increasing generalization by 2.026%, 3.081% and 2.351% respectively for ensembles trained with no augmentation, conventional augmentation, and Consistency Training. To reiterate, data augmentation contributed an average improvement of 9.00% and Consistency Training 11.73% over no augmentation. The differences in improvements between ensemble training methods was not however found to be statistically significant.

The findings as presented in Figure 4.13 do to some extent support the hypothesis that this improvement is a consequence of the fact that ensembles mitigate underspecification, as the greatest gains to generalization were achieved by models that initially exhibited high degrees of underspecification as quantified by the performance variability of the respective pipelines.

### 5.4.1 Impact

The use of ensembles was in this thesis proven to be a comparatively simple and reliable way of increasing generalization, albeit by a minor amount. It should also be noted that ensembles incur higher costs with regards to training time, time required for inference, and memory requirements. This needs to be weighed against the benefits, which as discussed are fairly marginal on average. It may for instance be the case that the computational resources spent training multiple predictors for use in an ensemble would

be better spent tuning the augmentation strategy if a OOD dataset is available. As the results in Section 4.3 show, the choice of augmentation strategy appears to have a more significant impact on generalization than the use of ensembles. Thus, the findings in this thesis suggest that testing N different augmentation strategies may be a better use of resources than training N identical predictors such that an ensemble can be implemented. Granted, this is difficult to say with certainty without exploring a larger diversity of augmentation strategies as discussed in Section 5.2.2.

The analysis of ensembles in the context of underspecification performed in Section 4.5.3 corroborated analyses in the literature. This also suggests that the possible returns from ensembles are limited however, and dependent on the landscape of the Bayesian posterior as discussed in Section 2.5.4.

### 5.4.2 Limitations

As mentioned in Section 4.5, the constituent predictors for each ensemble were sampled from the ten predictors trained for the purpose of the experiments in Section 4.3 and Section 4.4. As a result, the statistical significance of the findings are not necessarily robust. Thus, the experiments should ideally be repeated with an increased sample size, for instance N=50, such that ten ensembles could be constructed such that each ensemble consists of an independent set of predictors.

It should also be noted that the experiments in this thesis were performed only at one ensemble size - i.e, five models. This choice was informed by the literature, in particular the implementation of DivergentNet [88]. Ensembles may as such have a greater impact than expected, dependent on the returns from increasing the model counts. Following the Bayesian perspective as discussed in Section 2.5.4, increasing the model count may result in a better estimate of the Bayesian posterior, and thus lead to increased generalization.

The improvements from increasing ensemble size may however be limited. The performance of ensembles is after all bounded by the performance of perfect Bayesian marginalization. As shown in Figure 2.12, this will not necessarily constitute perfect generalization, as the predictions are in such a system weighted according to the likelihood that the given weight configuration is returned from the pipeline. Thus, if learning shortcuts is likely, Bayesian marginalization will primarily be predicting according to shortcut features. Investigating this may be an interesting direction of further study.

Finally, the method by which ensembles were implemented in this thesis - in particular, that the heatmap is thresholded with majority vote - may under-represent the potential utility of ensembles. By requiring that at least half of the constituent predictors are in consensus in order to consider a

given pixel as a positive prediction, some potentially insightful predictions may be discarded. A better alternative is to consider the heatmap as a whole, which in any case would be more informative in a clinical setting. Evaluation of ensemble models should thus ideally take this into account.

## 5.5 Impact in terms of Practical Utility

Though this thesis presents methods that constitute considerable improvements to generalization, the best performing system - namely the DeepLabV3+ ensemble trained with Consistency Training - would nevertheless not be particularly useful in practical settings when considered holistically.

This system achieved average mIoUs of 0.751 on CVC-ClinicDB, 0.683 on EndoCV2020, 0.523 on Etis-LaribDB, and 0.859 on Kvasir-SEG. Though this constitutes a considerable improvement over both of the "naive" pipelines - i.e single models trained with and without regular data augmentation - it is nonetheless not sufficiently generalizable for practical use. Ideally, there should be negligible differences between all four datasets, and though there is room for some degree of performance degradation, a system that exhibits a mean mIoU of 0.523 is not particularly useful and as discussed in Section 2.6 may actually cause more harm than good.

Thus, in spite of the aforementioned improvements, the pipeline as a whole is not in purely practical terms much better than any of the naive pipelines. More work is evidently required to achieve suitable levels of generalization. Some ideas for directions of further work towards this end will be discussed in Chapter 6.

## 5.6 Limitations of the Experimental Methodology

Though the experimental methodology used in this thesis afforded it a wide scope in addition to providing a suitable platform upon which generalizable methods could be investigated, it also had certain limitations. These will be discussed in the following sections.

### 5.6.1 Metrics selection

[73] As this thesis focused on the differences in performance between OOD and InD datasets, the only metrics that were considered was the mIoU and the C.StD of the mIoU. Though mIoU is a very popular metric for evaluating segmentation pipelines and is easy to interpret, it is not without its flaws [73]. It is for instance typically biased against small structures, as

these are affected to a greater degree by errors that in practical terms are comparatively minor. This can result in misleading mIoUs, in particular when the distribution of object sizes is wide, as is indeed often is the case with polyps datasets [1].

Ideally, more metrics should have been considered, for instance precision, recall, and perhaps even Segmentation Inconsistency Score (SIS), in order to paint a more complete picture of the performance of the tested methods.

### 5.6.2 Dataset Selection

This thesis considered three OOD datasets throughout all the experiments. Though this provides some indication of generalizability, ideally even more OOD datasets should have been used. For instance, though Etis-LaribDB was the most difficult of the datasets used in this thesis, the performance on this dataset does not necessarily reflect the worst-case performance in a clinical setting. Indeed, the extent to which the a given pipeline fails to generalize cannot be sufficiently anticipated [48] given current approaches to deep learning. It may easily be the case that the model performs even worse under certain clinical conditions. Without a larger sample of OOD datasets, there is a high degree of uncertainty involved as to the actual ability of the given systems to generalize. Though the low generalizability of the systems implemented in this thesis means that this has little practical bearing, any future research that reports generalizability of practical merit should concentrate a significant effort on assembling a large collection of OOD datasets.

Moreover, as briefly mentioned in Section 4.1, the methods presented in this thesis should ideally have been compared to the works presented in EndoCV2021 [3]. As the datasets used to evaluate the submissions was not available at the time of writing this thesis, however, this was not possible.

Synthetic tress-tests may also have been beneficial to implement. These stress-tests would however have to make use of a disjoint set of transformations than the augmentations used throughout the thesis.

### 5.6.3 Model Architectures

To validate that the impact of the proposed methods translated across models, and to determine the impact of model architectures, this thesis implemented five separate models - i.e DeepLabV3+, DD-DeepLabV3+, FPN, Unet, and TriUNet.

It can be argued that these models do not capture the diversity of segmentation models available, however. None of the models leverage any

form of attention or cascading, for instance, though both of these methods have been shown to increase generalization [27, 35].

Ideally, more models should have been implemented, including a selection of the non-ensemble models submitted to EndoCV2021[3] and/or other polyp-segmentation models[89]. Due to constraints both with regards to computational resources and implementation time, this was however infeasible.

## 5.7 Summary

This section presented an overview and discussion of the results and findings of each experiment as presented in Chapter 4. The impacts were discussed for each experiment, along with any identified limitations. The results were then put in context with the literature and considered in terms of viability of clinical deployment, before finally discussing limitations to the experimental methodology used across all experiments, i.e the selection of metrics, datasets, and models.

# Chapter 6

# Conclusion

## 6.1 Summary

The goal of this work was to develop novel methods of increasing the generalizability of deep learning models, as well as to survey the relative impacts of more conventional components of the deep learning pipeline. This was achieved as follows:

Chapter 2 provided an overview of deep learning, segmentation, and delved further into why such systems so readily fail to generalize, starting from first principles and analyzing the shortcomings of ERM. This was then connected to recent analyses of generalizability failure, including the notion of underspecification and shortcut learning. Finally, known methods of increasing generalization as presented in EndoCV2021 and elsewhere in the literature were then discussed and analyzed with respect to the established theory.

This was then in turn used to inform the development of the methods discussed in Chapter 3, including Consistency Training, the generative inpainter augmentation strategy, DD-DeepLabV3+ and a family of Consistency-trained ensemble models. Each of these methods were also discussed with respect to the theory explored in Chapter 2.

Several experiments were then conducted in Chapter 4 in order to ascertain the impact of the proposed methods: First, baseline generalizability metric were collected for five separate models. The findings supported the notion that larger models are more prone to generalizability failure, as demonstrated by the significant gap between the Unet and the TriUnet. The use of a secondary decoder in the DD-DeepLabV3+ model was shown to have negligible impact except for increased performance variability. It was hypothesized that this is due to the encoder already learning domain- and dataset-independent features, and thus that the additional parameters result in increased underspecification.

In the next experiment, data augmentation was shown to increase generalizability by a considerable margin. Synthetic augmentation via inpainting was shown to hamper this improvement when used in conjunction with regular augmentation.

The impact of Consistency Training was then tested and compared to conventional data augmentation and no augmentation. The results show that Consistency Training outperforms regular data augmentation by a considerable margin on all three OOD datasets when comparing across all models.

Finally, predictors trained in the previous experiments were then combined into ensembles and compared to one another. The results demonstrated the generalizability of ensemble-based methods, and that this can be traced to ensembles mitigating underspecification.

The results were then discussed in Chapter 5. The possible impacts of the findings in each experiment were considered, along with the limitations thereof. The overall practical utility of the best performing pipeline was then discussed, as well as the limitations of the experimental methodology.

Holistically, the findings in this thesis highlight that generalization remains a challenging problem, but that the development of generalizable methods is an endeavor ripe for further exploration. Consistency Training in particular seems to be a promising candidate for further research towards alleviating generalization failure. Secondly, the experimental methodology used in this thesis allowed for the identification of a number of variables previously unaccounted for in comparative research on generalizable methods, in particular with regards to the relative impacts of model architecture, augmentation, and ensemble networks. In particular, this thesis found that data augmentation had a greater impact than ensembles, which in turn had a greater impact than the choice of model architecture. Though this constitutes a good starting point towards developing a comprehensive understanding of impact of the many constituent components of the deep learning pipeline on generalizability, further foundational work is required, for instance by addressing some of the limitations as discussed in Chapter 5 or investigating some of the ideas for future work as will be presented in Section 6.3.

## 6.2 Contributions

The contributions of this can be summarized according to the research objectives laid out in Chapter 1:

**Objective 1**: *To leverage recent advances in the understanding of generalization failure to inform the development of novel methods of increasing the generalization*

*of deep learning systems for polyp-segmentation.*

This objective was achieved through the introduction of several novel methods, the most effective of which was shown to be Consistency Training. By reframing the problem of generalization as consistency across perturbations, Consistency Training was shown to increase generalizability by a considerable margin without the need for multiple training domains, in effect serving as a more generalizable alternative (or supplement) to data augmentation. This framework, and the potential improvements that can be made upon it as suggested in Chapter 5, shows good promise with regards to further increasing generalizability. The ensemble models consisting of predictors according to Consistency Training was also shown to increase generalization, outperforming conventionally trained ensembles. Though the remaining methods - i.e generative inpainting and DD-DeepLabV3+ - were proven to be ineffective, the analysis thereof nevertheless motivated a number of directions of further study.

**Objective 2:** *To synthesize recent work on generalizability and determine concretely the degree to which more conventional and well-established methods affect generalization.*

This objective was achieved by performing a quantitative analysis of the effect of the choice of model architecture, the use of data augmentation, and the use of ensembles on generalization. Though most of the findings corroborated the literature, there were a fair number of surprising results that warrant further investigation, in particular with regards to the impacts of the tested methods relative to one another. For one, the effect of multitask learning and generally the the choice of model architecture was in this thesis shown to be practically negligible. With the exception of TriUnet, every tested model exhibited practically identical performance. The use of ensemble-based model, though exhibiting statistically significant impact, resulted in somewhat marginal improvements on generalization, especially in comparison to the use of data augmentation and Consistency Training. As discussed in Chapter 5, this raises doubts as to the veracity of findings in other literature, where data augmentation is rarely accounted for when performing comparisons. Hopefully, the findings in this thesis demonstrate the need for a more structured approach to the design of experimental methodologies intended to analyze generalization, wherein the constituent components of the pipeline are sufficiently well controlled.

## 6.3 Future work

There are several directions of future research that may provide further insight into generalizability and generalization failure. This section will cover a number of these ideas.

### 6.3.1 Improving Consistency Training

As was shown in Chapter 4, Consistency Training is an effective means of increasing generalization. However, there is still room for further improvement and exploration. For instance, in this thesis consistency was expressed merely as the symmetric difference between the expected change in the output due to augmentation and the actual change due to augmentation. This, however, as discussed in Chapter 3, is largely agnostic to the augmentation being performed. However, the nature of these augmentations should be taken into account. If the image is subjected to a 90 degree rotation, for instance, the prediction would be considered perfectly consistent so long as the pixels corresponding to the polyps are rotated, and the incorrectly classified pixels remain unchanged. However, if the model instead learns to rotate all of the pixels - even those that are incorrectly classified - it may learn a more accurate representation of what constitutes consistent behavior. I.e, instead of expressing inconsistency as:

$$\bar{\mathcal{C}}(y, \hat{y}, a, \hat{a}) = \frac{\sum\{y \cap a \cap \hat{y} \cap \hat{a}\}}{\sum\{y \cup a \cup \hat{y} \cup \hat{a}\}}$$

One can adjust the expected change term $a \oplus y$ to $\hat{y} \oplus \epsilon(\hat{y})$ such that also incorrect predictions can be considered consistent so long as they change in accordance to the nature of the perturbation model $\epsilon(\cdot)$. The resulting loss function can then be expressed as:

$$\bar{\mathcal{C}}(\hat{y}, \hat{a}) = \sum \frac{\Theta(\hat{y}, \hat{a}, \hat{y}, \epsilon(\hat{y}))}{\bigcup(\hat{y}, \hat{a}, \epsilon(\hat{y}))}$$

Which is equivalent to:

$$\bar{\mathcal{C}}(\hat{y}, \hat{a}) = \sum \frac{\Theta(\hat{y}, \hat{a}, \epsilon(\hat{y}))}{\bigcup(\hat{y}, \hat{a}, \epsilon(\hat{y}))}$$

This also has the advantage of being independent of the labels themselves. This may alleviate complications that may arise as a consequence of poor and/or incomplete labeling which would otherwise affect what the models learn to associate with consistent behaviour.

In addition to improving the way by which consistency is quantified, there are several unexplored directions through which the training procedure itself could be further improved. The perturbation model, for instance, could be modified in any number of ways: one could for instance adversarially sample difficult augmentations based on the consistency score, and use these during training. One could also perform an ablation study to ascertain the impact of the perturbation models' constituent augmentation functions on generalization. It may for instance be the case that some of the augmentations used in the perturbation model used in this thesis hampered generalizability more than it facilitated it, though without a complete study this is impossible to say with any certainty.

One could also experiment with modulating the difficulty of the augmentations. In the experiments performed in this thesis, the augmentation difficulty was kept constant - i.e, the augmentation hyperparameters were capped to a specific range. However, it may be the case that gradually increasing the difficulty or modulate it according to some sort of annealing function could further improve the efficacy of Consistency Training.

Finally, using multiple perturbed images when computing inconsistency instead of just one may potentially further strengthen the generalizability of the learned features. In this thesis, the inconsistency term really only pertains to the inconsistency of the model with respect to the one change being applied to the perturbed input. It is possible to instead generate multiple perturbed inputs, each being transformed in a different manner, and then compute multiple inconsistency terms thereafter. This does require more memory, however, and may on certain hardware be infeasible unless the batch sizes are kept small.

### 6.3.2 Deep Denoising

In Consistency Training, the objective is to optimize for features that are consistent across perturbations such that the model learns invariance to distributional shifts that should not affect the causal structure of the problem. Though this as established increases generalizability, it may also be possible to use a DNN to simply preprocess the images such that OOD transformations or artifacts are accounted for. This is achieved elsewhere in the literature using generative models - for instance a CycleGAN [79], which maps the input data between domains prior to being given to the segmentation network. One could implement a similar system using Consistency Training through the use of a denoising network. The resulting pipeline is illustrated in Figure 6.1.

There are two main differences between this pipelines and more conventional deep denoising pipelines. First, the segmentation models are trained using Consistency Training. Second, the denoising network is incorporating SIL as a component of the loss function. There would in this case be two separate loss functions, one for each network. In theory, this should result in the denoiser learning to counteract the characteristics of the perturbations being applied that most negatively affect the consistency and thus the generalization of the segmentation models. Moreover, even if the denoiser performs poorly, the segmentation portion should be generalizable due to Consistency Training, which may even be improved as a result of whatever transforms the denoising network is performing, as these in effect could be treated as augmentations.

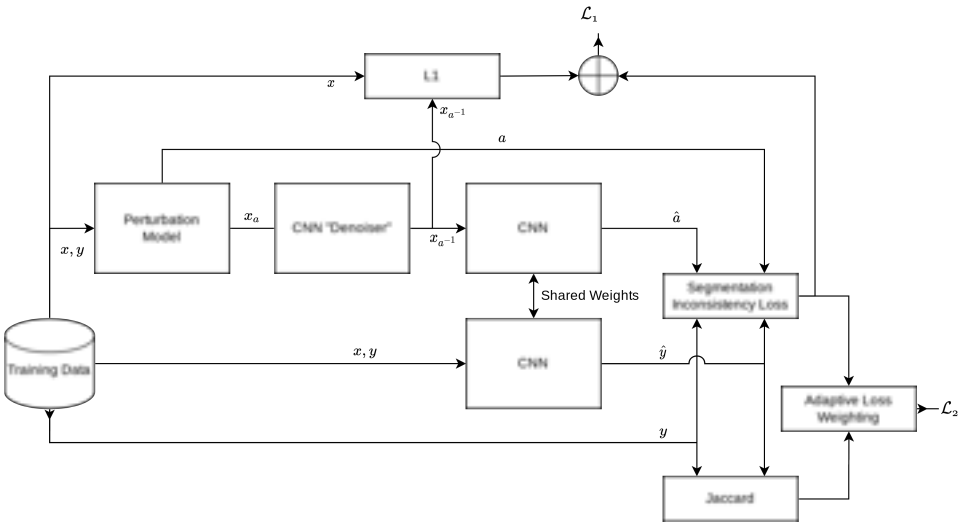


Figure 6.1: Consistency Preprocessing Pipeline

### 6.3.3 Further investigations of Multi-task learning

Chapter 3 introduced the dual-decoder DeepLabV3+, the intent of which was to increase generalization by constraining the space of latent representations that the model could leverage through a reconstruction decoder, thus in theory mitigating underspecification. As the results in Section 4.2.2 showed, however, the effect of this additional encoder was fairly limited when compared to the regular DeepLabV3+. It was hypothesized that this may be due to the encoder learning principally dataset-agnostic features and consequently primarily performing image compression regardless of what object the model is intended to segment.

This was to some extent corroborated by the analysis performed in Section 4.2.2, which showed equivalent image reconstruction performance across datasets in terms of L1 distance. Further research is however required, as these findings are only representative of one specific model trained in a limited number configurations. One possible direction is to implement a wider range of encoder-decoder models trained across multiple decoder tasks and datasets, and then investigate the latent spaces of the resulting predictors. If the predictor encoders indeed do encode primarily dataset-agnostic features independent of the decoder function, one would expect that one could simply switch encoders between predictors trained on different datasets, domains and tasks without significant performance degradation, at least after a few epochs of fine-tuning if for some reason the encoders have learned functionally different but practically comparable compression methods.

As discussed in Section 5.1.2, this behaviour might also be attributed to pretraining. Since the models used throughout this thesis were at

least partially pretrained on Imagenet, it might simply be the case that the encoders have learned to perform image compression as a direct consequence of the fact that this likely is the most conducive configuration to minimizing risk on the Imagenet dataset. In this case, the encoder may be in such a wide minimum that actually learning domain-specific features is unlikely even after training to segment polyps. Thus, testing the impact of different pretraining methods is also warranted.

Further investigating the impact of multitask learning beyond dual-decoder models is also warranted. One could for instance investigate decoupling segmentation into multiple stages, either through refinement stages or through attempting to learn unique complementary representations at different parts of the network through placing decoders at different depths of the network. Investigating the variance of the latent representation in these models across multiple runs of training and comparing these to the variance in single-task models may be interesting and further the understanding of what DNNs actually learn.

### 6.3.4 Further Investigations on Inpainting and Generative Modelling

The experiments in Section 4.3 showed that the use of an inpainter as implemented in this thesis harmed generalization when used in conjunction with conventional augmentations. Two hypotheses for why this is the case were suggested - either the inpainter simply does not perform to a sufficient standard conducive for use as augmentation, or the inpainter learned the distribution to such an extent that it increased the models' dataset bias.

To investigate this, it is possible to implement one of the more state-of-the-art inpainting architectures, for instance an inpainting generative multi-column network [92]. Additionally, analyzing the generated polyps via statistical means may also have some merit. The development of distance metrics to facilitate easier comparison between synthetic images to real images may for instance be worth looking into, as this might shed some light on the hypotheses as presented above.

### 6.3.5 Improving Ensembles through Diversity Search

Though ensembles as implemented in this thesis exhibit somewhat limited returns, leveraging a diversity of interpretations of the input data may have considerable merit towards increasing generalization and clinical utility. As the analysis in Figure 4.13 shows, there appears to be a positive relationship between generalizability and model-diversity that warrants further investigation. In particular, it may be the case that

ensembles consisting of predictors that are trained to explicitly encode differing features are more conducive to generalization than conventionally implemented ensembles. By explicitly optimizing for weight diversity, one might mitigate the tendency of typical ensembles to primarily consider weight configurations that exhibit higher posterior likelihoods.

This could for instance be achieved by training multiple instances of the same model concurrently, and incorporating some measure of the diversity of the learned feature maps across models into the loss function. A naive approach to this end could be to simply calculate the variance of each activation across every predictor. This is computationally expensive, however. A better approach is to select a subset of activations - such as the encoder outputs - and calculate the standard variance across the predictors just for this subset. An illustration of such a pipeline is provided in Figure 6.2.

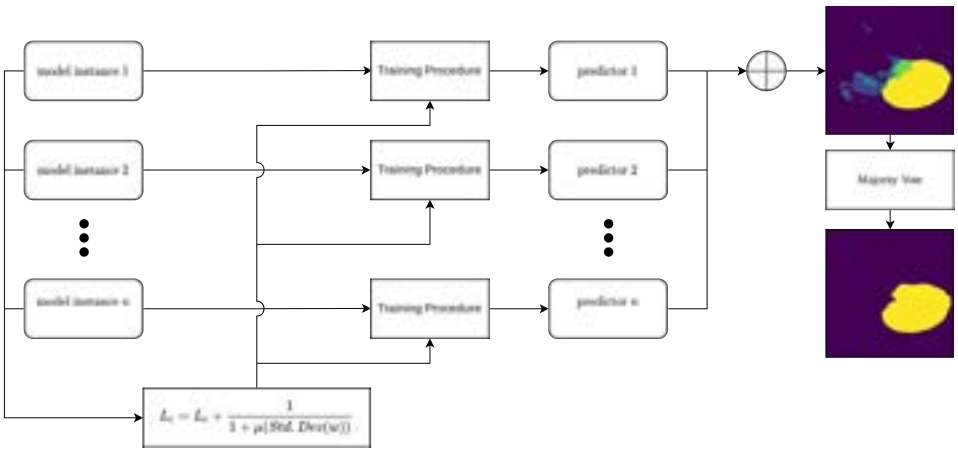


Figure 6.2: By adding a term corresponding to the mean standard deviation of weights, the models will learn maximally independent representations, and hence result in predictors with a larger diversity of learned features. This may mitigate underspecification to a greater extent, since this search would be less biased towards regions of the search landscape with high posterior probability.

This way, the ensemble will consist of predictors that encode a wider diversity of interpretations of the data than the predictors in conventional ensembles. This in turn provides a more complete perspective of the many possible interpretations a given model can learn. If this is proven to be the case, using the heatmaps from such an ensemble during screening may also be useful in clinical settings, as the clinician could then take all of these possible interpretations into account instead of trusting that a single predictor is encoding the right inductive biases.

# Bibliography


[1] Sharib Ali et al. *Assessing generalisability of deep learning-based polyp detection and segmentation methods through a computer vision challenge*. 2022. DOI: 10.48550/ARXIV.2202.12031. URL: https://arxiv.org/abs/2202.12031.

[2] Sharib Ali et al., eds. *EndoCV2020: 2nd International Workshop and Challenge on Computer Vision in Endoscopy*. Vol. 2595. Iowa, USA: CEUR Workshop Proceedings, 2020. URL: http://ceur-ws.org/Vol-2595/.

[3] Sharib Ali et al., eds. *Proceedings of the 3rd International Workshop and Challenge on Computer Vision in Endoscopy (EndoCV 2021) co-located with with the 17th IEEE International Symposium on Biomedical Imaging (ISBI 2021)*.

[4] Laith Alzubaidi et al. 'Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions'. In: *Journal of big Data* 8.1 (2021), pp. 1–74. DOI: https://doi.org/10.1186/s40537-021-00444-8. URL: https://link.springer.com/article/10.1186/s40537-021-00444-8.

[5] Martin Arjovsky et al. *Invariant Risk Minimization*. 2019. DOI: 10.48550/ARXIV.1907.02893. URL: https://arxiv.org/abs/1907.02893.

[6] Sanjeev Arora and Yi Zhang. *Do GANs actually learn the distribution? An empirical study*. 2017. DOI: 10.48550/ARXIV.1706.08224. URL: https://arxiv.org/abs/1706.08224.

[7] Sanjeev Arora et al. *Generalization and Equilibrium in Generative Adversarial Nets (GANs)*. 2017. DOI: 10.48550/ARXIV.1703.00573. URL: https://arxiv.org/abs/1703.00573.

[8] Razieh Baradaran and Hossein Amirkhani. 'Ensemble learning-based approach for improving generalization capability of machine reading comprehension systems'. In: *Neurocomputing* 466 (2021), pp. 229–242. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2021.08.095. URL: https://www.sciencedirect.com/science/article/pii/S0925231221012923.

[9] Ishita Barua et al. 'Artificial intelligence for polyp detection during colonoscopy: a systematic review and meta-analysis'. en. In: *Endoscopy* 53.3 (Mar. 2021), pp. 277–284. DOI: https://doi.org/10.1055/a-1201-7165. URL: https://pubmed.ncbi.nlm.nih.gov/32557490/.

[10] Emma Beede et al. 'A Human-Centered Evaluation of a Deep Learning System Deployed in Clinics for the Detection of Diabetic Retinopathy'. In: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. CHI '20. Honolulu, HI, USA: Association for Computing Machinery, 2020, 1–12. ISBN: 9781450367080. DOI: 10.1145/3313831.3376718. URL: https://doi.org/10.1145/3313831.3376718.

[11] Michael Beil et al. 'Ethical considerations about artificial intelligence for prognostication in intensive care'. In: *Intensive Care Medicine Experimental* 7.1 (2019), pp. 1–13. DOI: ttps://doi.org/10.1186/s40635-019-0286-6. URL: https://icm-experimental.springeropen.com/articles/10.1186/s40635-019-0286-6.

[12] Jorge Bernal et al. 'WM-DOVA maps for accurate polyp highlighting in colonoscopy: Validation vs. saliency maps from physicians'. In: *Computerized Medical Imaging and Graphics* 43 (2015), pp. 99–111. DOI: http://dx.doi.org/10.1016/j.compmedimag.2015.02.007. URL: https://polyp.grand-challenge.org/CVCClinicDB/.

[13] Battista Biggio et al. 'Evasion Attacks against Machine Learning at Test Time'. In: *Lecture Notes in Computer Science* (2013), 387–402. ISSN: 1611-3349. DOI: 10.1007/978-3-642-40994-3_25. URL: http://dx.doi.org/10.1007/978-3-642-40994-3_25.

[14] Raf Bisschops et al. 'Virtual chromoendoscopy (I-SCAN) detects more polyps in patients with Lynch syndrome: a randomized controlled crossover trial'. In: *Endoscopy* 49.04 (2017), pp. 342–350. DOI: https://doi.org/10.1055/s-0042-121005. URL: https://pubmed.ncbi.nlm.nih.gov/28107763/.

[15] Karam S. Boparai et al. 'Increased polyp detection using narrow band imaging compared with high resolution endoscopy in patients with hyperplastic polyposis syndrome.' In: *Endoscopy* 43 8 (2011), pp. 676–82. DOI: https://doi.org/10.1055/s-0030-1256447. URL: https://pubmed.ncbi.nlm.nih.gov/21811939/.

[16] Hanna Borgli et al. 'HyperKvasir, a comprehensive multi-class image and video dataset for gastrointestinal endoscopy'. In: *Scientific Data* 7.1 (2020), p. 283. ISSN: 2052-4463. DOI: 10.1038/s41597-020-00622-y. URL: https://doi.org/10.1038/s41597-020-00622-y.

[17] A. Buslaev et al. 'Albumentations: fast and flexible image augmentations'. In: *ArXiv e-prints* (2018). eprint: 1809.06839. URL: https://albumentations.ai/.

[18] Liang-Chieh Chen et al. *Rethinking Atrous Convolution for Semantic Image Segmentation*. 2017. DOI: 10.48550/ARXIV.1706.05587. URL: https://arxiv.org/abs/1706.05587.

[19] Corinna Cortes, Mehryar Mohri and Afshin Rostamizadeh. *L2 Regularization for Learning Kernels*. 2012. DOI: 10.48550/ARXIV.1205.2653. URL: https://arxiv.org/abs/1205.2653.

[20] Alexander D'Amour et al. *Underspecification Presents Challenges for Credibility in Modern Machine Learning*. 2020. DOI: 10.48550/ARXIV.2011.03395. URL: https://arxiv.org/abs/2011.03395.

[21] Joel Dapello et al. 'Simulating a Primary Visual Cortex at the Front of CNNs Improves Robustness to Image Perturbations'. In: *bioRxiv* (2020). DOI: 10.1101/2020.06.16.154542. eprint: https://www.biorxiv.org/content/early/2020/06/17/2020.06.16.154542.full.pdf. URL: https://www.biorxiv.org/content/early/2020/06/17/2020.06.16.154542.

[22] Aminu Da'u and Naomie Salim. 'Recommendation system based on deep learning methods: a systematic review and new directions'. In: *Artificial Intelligence Review* 53.4 (2020), pp. 2709–2748. DOI: https://doi.org/10.1007/s10462-019-09744-1. URL: https://link.springer.com/article/10.1007/s10462-019-09744-1.

[23] Logan Engstrom et al. *Exploring the Landscape of Spatial Robustness*. 2017. DOI: 10.48550/ARXIV.1712.02779. URL: https://arxiv.org/abs/1712.02779.

[24] Paul F Engstrom et al. 'Colon cancer'. In: *Journal of the National Comprehensive Cancer Network* 7.8 (2009), pp. 778–831. URL: https://pubmed.ncbi.nlm.nih.gov/17977501/.

[25] Dumitru Erhan et al. 'Why Does Unsupervised Pre-training Help Deep Learning?' In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Yee Whye Teh and Mike Titterington. Vol. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy: PMLR, May 2010, pp. 201–208. URL: https://proceedings.mlr.press/v9/erhan10a.html.

[26] Ivan Evtimov et al. 'Robust Physical-World Attacks on Machine Learning Models'. In: *CoRR* abs/1707.08945 (2017). arXiv: 1707.08945. URL: http://arxiv.org/abs/1707.08945.

[27] Adrian Galdran, Gustavo Carneiro and Miguel A. González Ballester. 'Multi-Center Polyp Segmentation with Double Encoder-Decoder Networks'. In: *Proceedings of the 3rd International Workshop and Challenge on Computer Vision in Endoscopy (EndoCV 2021) co-located with with the 17th IEEE International Symposium on Biomedical Imaging (ISBI 2021)*. 2021, pp. 9–16. URL: http://ceur-ws.org/Vol-2886/paper1.pdf.

[28] Robert Geirhos et al. *ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness*. 2018. DOI: 10.48550/ARXIV.1811.12231. URL: https://arxiv.org/abs/1811.12231.

[29] Robert Geirhos et al. 'Shortcut learning in deep neural networks'. In: *Nature Machine Intelligence* 2.11 (Nov. 2020), 665–673. ISSN: 2522-5839. DOI: 10.1038/s42256-020-00257-z. URL: http://dx.doi.org/10.1038/s42256-020-00257-z.

[30] Raman Ghimirea, Sahadev Poudelb and Sang-Woong Leec. 'An Augmentation Strategy with Lightweight Network for Polyp Segmentation'. In: *Proceedings of the 3rd International Workshop and Challenge on Computer Vision in Endoscopy (EndoCV 2021) co-located with with the 17th IEEE International Symposium on Biomedical Imaging (ISBI 2021)*. 2021. URL: http://ceur-ws.org/Vol-2886/paper4.pdf.

[31] Tejas Gokhale et al. *Generalized but not Robust? Comparing the Effects of Data Modification Methods on Out-of-Domain Generalization and Adversarial Robustness*. 2022. DOI: 10.48550/ARXIV.2203.07653. URL: https://arxiv.org/abs/2203.07653.

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

[33] Ian J. Goodfellow et al. *Generative Adversarial Networks*. 2014. DOI: 10.48550/ARXIV.1406.2661. URL: https://arxiv.org/abs/1406.2661.

[34] Sven Gowal et al. *Improving Robustness using Generated Data*. 2021. DOI: 10.48550/ARXIV.2110.09468. URL: https://arxiv.org/abs/2110.09468.

[35] Ran Gu et al. 'Domain Composition and Attention for Unseen-Domain Generalizable Medical Image Segmentation'. In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*. Ed. by Marleen de Bruijne et al. Cham: Springer International Publishing, 2021, pp. 241–250. ISBN: 978-3-030-87199-4. DOI: https://doi.org/10.1007/978-3-030-87199-4_23.

[36] Mahmood Haithami et al. 'An Embedded Recurrent Neural Network-based Model for Endoscopic Semantic Segmentation'. In: *Proceedings of the 3rd International Workshop and Challenge on Computer Vision in Endoscopy (EndoCV 2021) co-located with with the 17th IEEE International Symposium on Biomedical Imaging (ISBI 2021)*. 2021. URL: http://ceur-ws.org/Vol-2886/paper6.pdf.

[37] Dan Hendrycks and Thomas Dietterich. *Benchmarking Neural Network Robustness to Common Corruptions and Perturbations*. 2019. arXiv: 1903.12261 [cs.LG].

[38] D Heresbach et al. 'Miss rate for colorectal neoplastic polyps: a prospective multicenter study of back-to-back video colonoscopies'. en. In: *Endoscopy* 40.4 (Apr. 2008), pp. 284–290. DOI: https://doi.org/10.1055/s-2007-995618. URL: https://pubmed.ncbi.nlm.nih.gov/18389446/.

[39] Steven Hicks et al. 'The EndoTect 2020 Challenge: Evaluation and Comparison of Classification, Segmentation and Inference Time for Endoscopy'. In: Feb. 2021, pp. 263–274. ISBN: 978-3-030-68792-2. DOI: 10.1007/978-3-030-68793-9_18.

[40] Geoffrey E. Hinton et al. *Improving neural networks by preventing co-adaptation of feature detectors*. 2012. DOI: 10.48550/ARXIV.1207.0580. URL: https://arxiv.org/abs/1207.0580.

[41] Ayoung Honga et al. 'Deep Learning Model Generalization with Ensemble in Endoscopic Images'. In: *Proceedings of the 3rd International Workshop and Challenge on Computer Vision in Endoscopy (EndoCV 2021) co-located with with the 17th IEEE International Symposium on Biomedical Imaging (ISBI 2021)*. 2021, pp. 80–89. URL: http://ceur-ws.org/Vol-2886/paper8.pdf.

[42] Hossein Hosseini, Baicen Xiao and Radha Poovendran. *Google's Cloud Vision API Is Not Robust To Noise*. 2017. arXiv: 1704.05051 [cs.CV].

[43] Jian Huang, Junyi Chai and Stella Cho. 'Deep learning in finance and banking: A literature review and classification'. In: *Frontiers of Business Research in China* 14.1 (2020), pp. 1–24. DOI: https://doi.org/10.1186/s11782-020-00082-6. URL: https://fbr.springeropen.com/articles/10.1186/s11782-020-00082-6.

[44] Andrew Ilyas et al. *Adversarial Examples Are Not Bugs, They Are Features*. 2019. DOI: 10.48550/ARXIV.1905.02175. URL: https://arxiv.org/abs/1905.02175.

[45] Andrew Ilyas et al. 'Black-box Adversarial Attacks with Limited Queries and Information'. In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 2137–2146. URL: https://proceedings.mlr.press/v80/ilyas18a.html.

[46] Sergey Ioffe and Christian Szegedy. 'Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift'. In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 448–456. URL: https://proceedings.mlr.press/v37/ioffe15.html.

[47] Iyad Issa and Malak Noureddine. 'Colorectal cancer screening: An updated review of the available options Iyad A Issa, Malak Noureddine'. In: *World Journal of Gastroenterology* 23 (July 2017), p. 5086. DOI: 10.3748/wjg.v23.i28.5086. URL: https://www.researchgate.net/publication/318661385_Colorectal_cancer_screening_An_updated_review_of_the_available_options_Iyad_A_Issa_Malak_Noureddine.

[48] Alon Jacovi et al. *Formalizing Trust in Artificial Intelligence: Prerequisites, Causes and Goals of Human Trust in AI*. 2020. DOI: 10.48550/ARXIV.2010.07487. URL: https://arxiv.org/abs/2010.07487.

[49] Debesh Jha et al. *Kvasir-SEG: A Segmented Polyp Dataset*. 2019. DOI: 10.48550/ARXIV.1911.07069. URL: https://arxiv.org/abs/1911.07069.

[50] Tero Karras et al. *Analyzing and Improving the Image Quality of StyleGAN*. 2019. DOI: 10.48550/ARXIV.1912.04958. URL: https://arxiv.org/abs/1912.04958.

[51] Jacob Kauffmann et al. *The Clever Hans Effect in Anomaly Detection*. 2020. arXiv: 2006.10609 [cs.LG].

[52] Amit Kaushal, Russ Altman and Curt Langlotz. 'Geographic Distribution of US Cohorts Used to Train Deep Learning Algorithms'. In: *JAMA* 324.12 (Sept. 2020), pp. 1212–1213. ISSN: 0098-7484. DOI: 10.1001/jama.2020.12067. eprint: https://jamanetwork.com/journals/jama/articlepdf/2770833/jama_kaushal_2020_ld_200073_1600712104.82262.pdf. URL: https://doi.org/10.1001/jama.2020.12067.

[53] Taehun Kim, Hyemin Lee and Daijin Kim. 'UACANet: Uncertainty Augmented Context Attention for Polyp Segmentation'. In: *Proceedings of the 29th ACM International Conference on Multimedia*. ACM, Oct. 2021. DOI: 10.1145/3474085.3475375. URL: https://doi.org/10.1145%2F3474085.3475375.

[54] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. DOI: 10.48550/ARXIV.1412.6980. URL: https://arxiv.org/abs/1412.6980.

[55] Anders Krogh and John Hertz. 'A Simple Weight Decay Can Improve Generalization'. In: 4 (1991). Ed. by J. Moody, S. Hanson and R.P. Lippmann. URL: https://proceedings.neurips.cc/paper/1991/file/8eefcfdf5990e441f0fb6f3fad709e21-Paper.pdf.

[56] Ranjit Kumar. *Research methodology: A step-by-step guide for beginners*. Sage, 2018.

[57] Nikhil KumarTomar et al. 'Improving generalizibilty in polyp segmentation using ensemble convolutional neural network'. In: vol. 2886. CEUR Workshop Proceedings, 2021. URL: http://ceur-ws.org/Vol-2886/paper5.pdf.

[58] Agostina J. Larrazabal et al. 'Gender imbalance in medical imaging datasets produces biased classifiers for computer-aided diagnosis'. In: *Proceedings of the National Academy of Sciences*. Vol. 117. 23. National Academy of Sciences, 2020, pp. 12592–12594. DOI: 10.1073/pnas.1919012117. eprint: https://www.pnas.org/content/117/23/12592.full.pdf. URL: https://www.pnas.org/content/117/23/12592.

[59] Fahad Lateef and Yassine Ruichek. 'Survey on semantic segmentation using deep learning techniques'. In: *Neurocomputing* 338 (2019), pp. 321–348. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2019.02.003. URL: https://www.sciencedirect.com/science/article/pii/S092523121930181X.

[60] Phuc H. Le-Khac, Graham Healy and Alan F. Smeaton. 'Contrastive Representation Learning: A Framework and Review'. In: *IEEE Access* 8 (2020), pp. 193907–193934. DOI: 10.1109/access.2020.3031549. URL: https://doi.org/10.1109/access.2020.3031549.

[61] A Leslie et al. 'The colorectal adenoma-carcinoma sequence'. en. In: *Br. J. Surg.* 89.7 (July 2002), pp. 845–860. DOI: https://doi.org/10.1046/j.1365-2168.2002.02120.x. URL: https://pubmed.ncbi.nlm.nih.gov/12081733/.

[62] Tsung-Yi Lin et al. 'Feature Pyramid Networks for Object Detection'. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[63] R. Lippmann. 'An introduction to computing with neural nets'. In: *IEEE ASSP Magazine* 4.2 (1987), pp. 4–22. DOI: 10.1109/MASSP.1987.1165576.

[64] Zachary C. Lipton, John Berkowitz and Charles Elkan. 'A Critical Review of Recurrent Neural Networks for Sequence Learning'. In: (2015). DOI: 10.48550/ARXIV.1506.00019. URL: https://arxiv.org/abs/1506.00019.

[65] Jun Ma et al. 'Loss Odyssey in Medical Image Segmentation'. In: *Medical Image Analysis* 71 (2021), p. 102035. DOI: https://doi.org/10.1016/j.media.2021.102035. URL: https://www.sciencedirect.com/science/article/pii/S1361841521000815.

[66] Shervin Minaee et al. 'Image Segmentation Using Deep Learning: A Survey'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021), pp. 1–1. DOI: 10.1109/TPAMI.2021.3059968.

[67] Mohammad Momeny et al. 'Learning-to-augment strategy using noisy and denoised data: Improving generalizability of deep CNN for the detection of COVID-19 in X-ray images'. In: *Computers in Biology and Medicine* 136 (2021), p. 104704. ISSN: 0010-4825. DOI: https://doi.org/10.1016/j.compbiomed.2021.104704. URL: https://www.sciencedirect.com/science/article/pii/S0010482521004984.

[68] Roman Novak et al. *Sensitivity and Generalization in Neural Networks: an Empirical Study*. 2018. DOI: 10.48550/ARXIV.1802.08760. URL: https://arxiv.org/abs/1802.08760.

[69] Ziad Obermeyer et al. 'Dissecting racial bias in an algorithm used to manage the health of populations'. In: *Science* 366.6464 (2019), pp. 447–453. DOI: https://doi.org/10.1126/science.aax2342. URL: https://pubmed.ncbi.nlm.nih.gov/31649194/.

[70] Daniel W. Otter, Julian R. Medina and Jugal K. Kalita. 'A Survey of the Usages of Deep Learning in Natural Language Processing'. In: (2018). DOI: 10.48550/ARXIV.1807.10854. URL: https://arxiv.org/abs/1807.10854.

[71] Deepak Pathak et al. 'Context Encoders: Feature Learning by Inpainting'. In: (2016). DOI: 10.48550/ARXIV.1604.07379. URL: https://arxiv.org/abs/1604.07379.

[72] Harry A. Pierson and Michael S. Gashler. 'Deep Learning in Robotics: A Review of Recent Research'. In: (2017). DOI: 10.48550/ARXIV.1707.07217. URL: https://arxiv.org/abs/1707.07217.

[73] Annika Reinke et al. *Common Limitations of Image Processing Metrics: A Picture Story*. 2021. DOI: 10.48550/ARXIV.2104.05642. URL: https://arxiv.org/abs/2104.05642.

[74] D K Rex et al. 'Colonoscopic miss rates of adenomas determined by back-to-back colonoscopies'. en. In: *Gastroenterology* 112.1 (Jan. 1997), pp. 24–28. DOI: https://doi.org/10.5009/gnl.2012.6.1.64.

[75] Alexander Robey, Hamed Hassani and George J. Pappas. *Model-Based Robust Deep Learning: Generalizing to Natural, Out-of-Distribution Data*. 2020. arXiv: 2005.10247 [cs.LG].

[76] Olaf Ronneberger, Philipp Fischer and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. DOI: 10.48550/ARXIV.1505.04597. URL: https://arxiv.org/abs/1505.04597.

[77] Sebastian Ruder. 'An overview of gradient descent optimization algorithms'. In: (2016). DOI: 10.48550/ARXIV.1609.04747. URL: https://arxiv.org/abs/1609.04747.

[78] Sebastian Ruder. *An Overview of Multi-Task Learning in Deep Neural Networks*. 2017. DOI: 10.48550/ARXIV.1706.05098. URL: https://arxiv.org/abs/1706.05098.

[79] Veit Sandfort et al. 'Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks'. In: *Scientific Reports* 9 (Nov. 2019). DOI: 10.1038/s41598-019-52737-x.

[80] Thomas H. Sanford et al. 'Data Augmentation and Transfer Learning to Improve Generalizability of an Automated Prostate Segmentation Model'. In: *American Journal of Roentgenology* 215.6 (2020). PMID: 33052737, pp. 1403–1410. DOI: 10.2214/AJR.19.22347. eprint: https://doi.org/10.2214/AJR.19.22347. URL: https://doi.org/10.2214/AJR.19.22347.

[81] Bernhard Schölkopf. *Causality for Machine Learning*. 2019. DOI: 10.48550/ARXIV.1911.10500. URL: https://arxiv.org/abs/1911.10500.

[82] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[83] Janelle Shane. 'Do neural nets dream of electric sheep?' In: *AI Wierdness* (2018). URL: https://www.aiweirdness.com/do-neural-nets-dream-of-electric-18-03-02/.

[84] Dinggang Shen, Guorong Wu and Heung-Il Suk. 'Deep Learning in Medical Image Analysis'. In: *Annual Review of Biomedical Engineering* 19.1 (2017). PMID: 28301734, pp. 221–248. DOI: 10.1146/annurev-bioeng-071516-044442. eprint: https://doi.org/10.1146/annurev-bioeng-071516-044442. URL: https://doi.org/10.1146/annurev-bioeng-071516-044442.

[85] Ashish Shrivastava et al. 'Learning from Simulated and Unsupervised Images through Adversarial Training'. In: arXiv, 2016. DOI: 10.48550/ARXIV.1612.07828. URL: https://arxiv.org/abs/1612.07828.

[86] Juan Silva et al. 'Toward embedded detection of polyps in wce images for early diagnosis of colorectal cancer'. In: *International journal of computer assisted radiology and surgery* 9.2 (2014), pp. 283–293. DOI: https://doi.org/10.1007/s11548-013-0926-3.

[87] David Silver et al. 'Mastering the game of Go with deep neural networks and tree search'. In: *nature* 529.7587 (2016), pp. 484–489. DOI: https://doi.org/10.1038/nature16961. URL: https://www.nature.com/articles/nature16961.

[88] Vajira Thambawita et al. 'DivergentNets: Medical Image Segmentation by Network Ensemble'. In: *Proceedings of the 3rd International Workshop and Challenge on Computer Vision in Endoscopy (EndoCV 2021) co-located with with the 17th IEEE International Symposium on Biomedical Imaging (ISBI 2021)*. 2021, pp. 27–38. URL: https://arxiv.org/abs/2107.00283.

[89] Nikhil Kumar Tomar et al. 'DDANet: Dual Decoder Attention Network for Automatic Polyp Segmentation'. In: *Pattern Recognition. ICPR International Workshops and Challenges*. Ed. by Alberto Del Bimbo et al. Cham: Springer International Publishing, 2021, pp. 307–314. ISBN: 978-3-030-68793-9. DOI: 10.1007/978-3-030-68793-9_23. URL: https://link.springer.com/chapter/10.1007/978-3-030-68793-9_23.

[90] Ashish Vaswani et al. 'Attention Is All You Need'. In: (2017). DOI: 10.48550/ARXIV.1706.03762. URL: https://arxiv.org/abs/1706.03762.

[91] Athanasios Voulodimos et al. 'Deep Learning for Computer Vision: A Brief Review'. In: *Computational Intelligence and Neuroscience* 2018 (Feb. 2018), pp. 1–13. DOI: 10.1155/2018/7068349.

[92] Yi Wang et al. 'Image Inpainting via Generative Multi-column Convolutional Neural Networks'. In: (2018). DOI: 10.48550/ARXIV.1810.08771. URL: https://arxiv.org/abs/1810.08771.

[93] Andrew Gordon Wilson. *The Case for Bayesian Deep Learning*. 2020. DOI: 10.48550/ARXIV.2001.10995. URL: https://arxiv.org/abs/2001.10995.

[94] Andrew Gordon Wilson and Pavel Izmailov. *Bayesian Deep Learning and a Probabilistic Perspective of Generalization*. 2020. DOI: 10.48550/ARXIV.2002.08791. URL: https://arxiv.org/abs/2002.08791.

[95] Sidney J. Winawer et al. 'Prevention of Colorectal Cancer by Colonoscopic Polypectomy'. In: *New England Journal of Medicine* 329.27 (1993). PMID: 8247072, pp. 1977–1981. DOI: 10.1056/NEJM199312303292701. eprint: https://doi.org/10.1056/NEJM199312303292701. URL: https://doi.org/10.1056/NEJM199312303292701.

[96] Julia Winkler et al. 'Association Between Surgical Skin Markings in Dermoscopic Images and Diagnostic Performance of a Deep Learning Convolutional Neural Network for Melanoma Recognition'. In: *JAMA Dermatology* 155 (Aug. 2019). DOI: 10.1001/jamadermatol.2019.1735.

[97] Pavel Yakubovskiy. *Segmentation Models Pytorch*. https://github.com/qubvel/segmentation_models.pytorch. 2020.

[98] ChengHui Yua, JiangPeng Yana and Xiu Lia. 'Parallel Res2Net-based Network with Reverse Attention for Polyp Segmentation'. In: *Proceedings of the 3rd International Workshop and Challenge on Computer Vision in Endoscopy (EndoCV 2021) co-located with with the 17th IEEE International Symposium on Biomedical Imaging (ISBI 2021)*. 2021, pp. 17–26. URL: http://ceur-ws.org/Vol-2886/paper2.pdf.

[99] John R. Zech et al. 'Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study'. In: *PLOS Medicine* 15.11 (Nov. 2018), pp. 1–17. DOI: 10.1371/journal.pmed.1002683. URL: https://doi.org/10.1371/journal.pmed.1002683.

[100] Chiyuan Zhang et al. *Understanding deep learning requires rethinking generalization*. 2016. DOI: 10.48550/ARXIV.1611.03530. URL: https://arxiv.org/abs/1611.03530.

# Appendix A

# Code Access

All relevant code and data can be found on the GitHub repository: https://github.com/BirkTorpmannHagen/Master

# Appendix B

# p-values

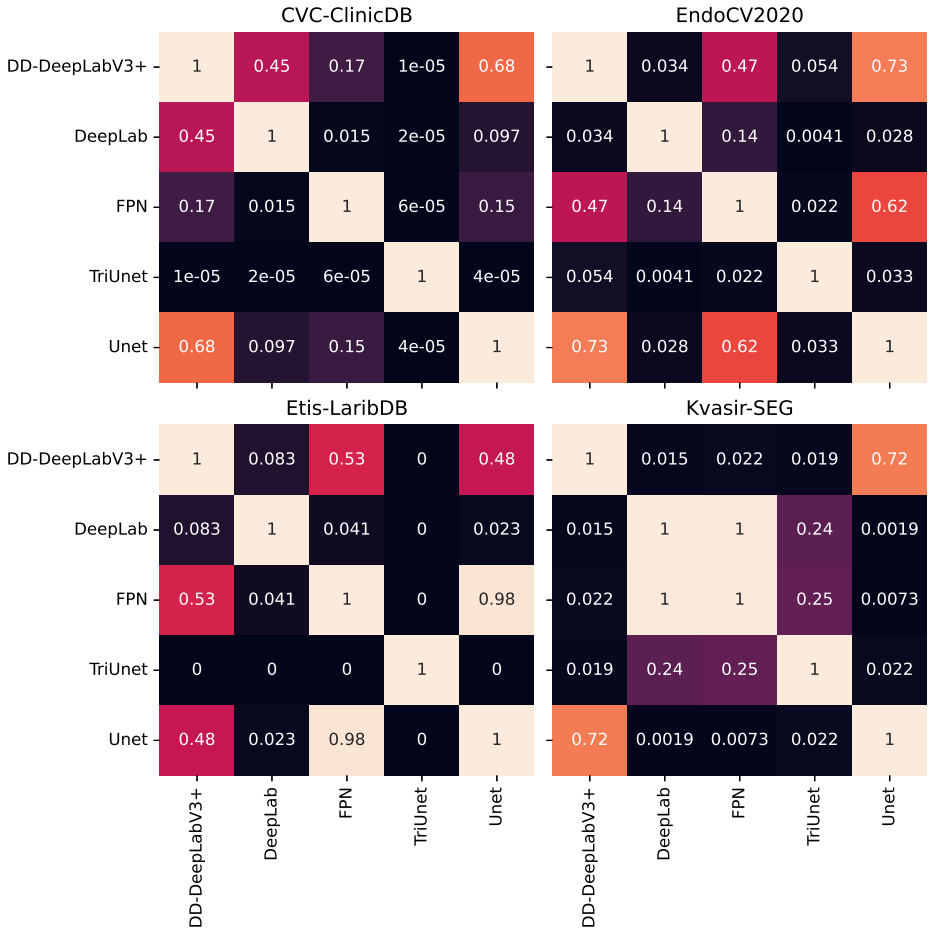


Figure B.1: Two-sided independent t-test p-values between models for all datasets

| Model | CVC-ClinicDB | EndoCV2020 | Etis-LaribDB | Kvasir-SEG |
|---|---|---|---|---|
| DD-DeepLabV3+ | 0.04454 | 0.95857 | 0.12809 | 0.30201 |
| DeepLab | **0.0096** | 0.08898 | 0.11401 | 0.31065 |
| FPN | 0.13769 | 0.95284 | 0.17806 | 0.16613 |
| TriUnet | 0.13412 | 0.31111 | 0.19913 | 0.91489 |
| Unet | 0.01069 | 0.15406 | 0.02715 | 0.36489 |

Table B.1: p-values for each model and dataset between the IoUs of the given models trained with versus when trained with conventional data augmentation versus models trained with the inpainter as a component of the data augmentation strategy

| Dataset | U-Statistic | p-Value |
|---|---|---|
| Kvasir-SEG | 763.0 | 0.15972 |
| Etis-LaribDB | 545.0 | 0.00163 |
| EndoCV2020 | 851.0 | 0.4169 |
| CVC-ClinicDB | 520.0 | 0.00077 |

Table B.2: Results from Mann-Whitney U-test for each dataset when comparing the mIoUs of all models trained with conventional data augmentation versus models trained with the inpainter as a component of the data augmentation strategy

| Dataset | U-Statistic | p-Value |
|---|---|---|
| Kvasir-SEG | 1066.0 | 0.10293 |
| Etis-LaribDB | 624.0 | 0.00001 |
| CVC-ClinicDB | 751.0 | 0.00029 |
| EndoCV2020 | 774.0 | 0.00052 |

Table B.3: Results from a Mann-Whitney U-test for each dataset when comparing the mIoUs across models for Consistency Training vs conventional data augmentation

| Model | CVC-ClinicDB | EndoCV2020 | Etis-LaribDB | Kvasir-SEG |
|---|---|---|---|---|
| DD-DeepLabV3+ | 0.014 | 0.985 | 0.083 | 0.170 |
| DeepLab | 0.029 | 0.901 | **0.003** | 0.444 |
| FPN | **0.004** | 0.038 | **0.005** | 0.939 |
| TriUnet | 0.211 | 0.024 | 0.141 | 0.330 |
| Unet | **0.000** | **0.001** | **0.006** | 0.899 |

Table B.4: p-values for each model and dataset between the mIoUs of the given models trained with consistency training versus when trained with data augmentation

| Training method | CVC-ClinicDB | EndoCV2020 | Etis-LaribDB | Kvasir-SEG |
|---|---|---|---|---|
| No Augmentation | 0.000 | 0.000 | 0.006 | 0.000 |
| Conventional Augmentation | 0.000 | 0.000 | 0.001 | 0.000 |
| Consistency Training | 0.000 | 0.000 | 0.003 | 0.000 |

Table B.5: p-values from a Mann-Whitney U-test for each dataset and training method when comparing the mIoU of ensembles vs. the mIoU across its constituent models.

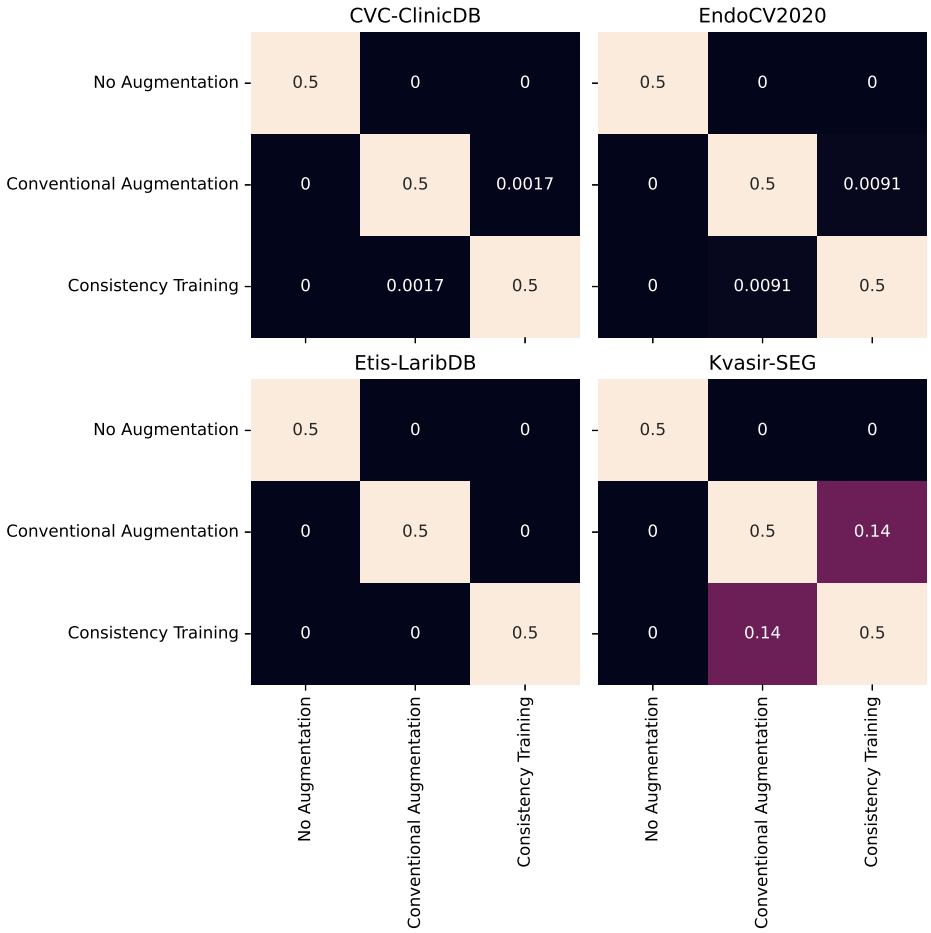


Figure B.2: Results from Mann-Whitney U-test for each dataset when comparing the mIoUs across models for the three training methods.

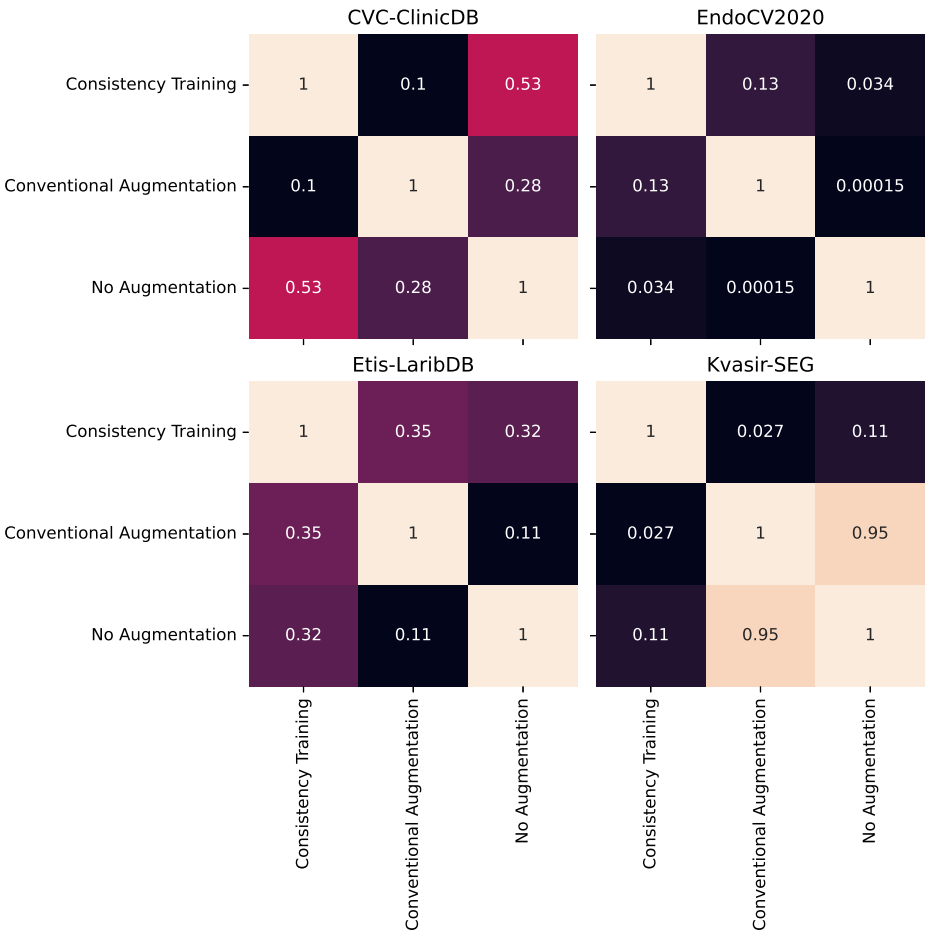


Figure B.3: Results from an independent-sample two-sided t-test when comparing the relative improvements across across models for the three training methods.

# Appendix C

# Non-weighted Consistency Training

Figure C.1: When the consistency term is not modulated dynamically, the model can quickly learn to predict artifacts around the edges of the image. As polyps can rarely be found in these regions, the consistency term is minimized by predicting consistently wrong predictions where there typically are not polyps.

# Appendix D

# Paper submitted to NeurIPS2022

# Segmentation Consistency Training: Out-of-Distribution Generalization for Medical Image Segmentation

**Birk Torpmann-Hagen**
University of Oslo
birk.torpmann.hagen@gmail.com

**Vajira Thambawita**
SimulaMet
vajira@simula.no

**Kyrre Glette**
University of Oslo
kyrrehg@ifi.uio.no

**Pål Halvorsen**
SimulaMet
paalh@simula.no

**Michael A. Riegler**
SimulaMet
michael@simula.no

## Abstract

Generalizability is seen as one of the major challenges in deep learning, in particular in the domain of medical imaging, where a change of hospital or in imaging routines can lead to a complete failure of a model. To tackle this, we introduce **Consistency Training**, a training procedure and alternative to data augmentation based on maximizing models' prediction consistency across augmented and unaugmented data in order to facilitate better out-of-distribution generalization. To this end, we develop a novel region-based segmentation loss function called Segmentation Inconsistency Loss (SIL), which considers the differences between pairs of augmented and unaugmented predictions and labels. We demonstrate that Consistency Training outperforms conventional data augmentation on several out-of-distribution datasets on polyp segmentation, a popular medical task.

## 1 Introduction

The last decade or so has seen a veritable revolution in Artificial Intelligence (AI). This has in large part been spearheaded by advancements in deep learning, the remarkable performance of which has rendered more conventional approaches practically obsolete. Recent work has however highlighted that Deep Neural Networks (DNNs) are highly prone to exhibiting significant reductions in performance when deployed in practical settings or otherwise Out of Distribution (OOD) data, in spite of the fact that they readily exhibit high performance when evaluated on previously unseen subsets of the training data [9, 11, 15, 18]. This is referred to as *generalization failure*.

Recent analyses attribute generalization failure to a structural misalignment between the features that a given model learns through Empirical Risk Minimization (ERM) and the causal structure which it ideally should encode [4, 11, 19, 26]. Generally, this misalignment occurs as a result of the predictor learning spurious or otherwise causally unrepresentative features that nonetheless perform well within the training distribution. This is often referred to as *shortcut learning* [11] or the *Clever Hans effect* [21]. This behaviour is of course made evident as soon as the predictor is exposed to any form of distributional shift which breaks these shortcuts, at which point it will fail to generalize. These distributional shifts can range in magnitude, from common corruptions such as noise or blurs [15] or spatial transforms [10], to practically imperceptible perturbations, typically exemplified by adversarial attacks [6], or as will be shown in this work; simply collecting data from different centers [29]. ERM does not and cannot guarantee invariance to these sorts of distributional shifts, as

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

it assumes that the distribution of the training data is Independent and Identically Distributed (IID) to the true distribution [13].

Closely related to shortcut learning is underspecification [9]. A machine learning pipeline can be considered underspecified when it can return any number of risk-equivalent predictors when evaluated on an IID holdout set, dependent only on the random variables used within the training procedure - i.e dropout, weight initialization, and so on. Even with identical hyperparameters, a given training procedure can return any number of predictors, each having learned different patterns within the dataset. One predictor may have learned one shortcut, another may have learned a different shortcut, and the next may actually have learned features that correspond to the causal structure it is intended to learn. With ERM, and in particular with In-Distribution (InD)-oriented evaluation procedures, these are all erroneously considered equivalent.

EndoCV2021 provided an opportunity to investigate generalization failure and means by which to counteract them in the context of detection- and segmentation of colorectal polyps via a competition [3]. Though several teams made good progress towards increasing generalizability, the organizers' review of the submissions [1] highlighted that every submitted model nevertheless exhibited significant performance reductions on the provided OOD datasets. Moreover, though a multitude of methods and approaches were tested, many of which did indeed benefit generalizability, few methods stood out as having the potential for significant further development.

To address these shortcomings, we introduce **Consistency Training**. We re-frame the problem of learning generalizable features into a matter of learning to *not* learn spurious features. This framework requires a *perturbation model*, which we in this work implement as simple data augmentation, and a differentiable quantity that represents the consistency of the predictions across perturbed and unperturbed inputs images, which we implement as *Segmentation Inconsistency Loss (SIL)*, a Jaccard-like loss function that quantifies the degree to which the segmentation probability maps exhibit unwarranted change after the input is perturbed. This loss function is then used in conjunction with a task-specific loss, in this work Jaccard loss. To increase the stability of the training routine, we also implement a dynamic weighting procedure for the two constituent components of the overall loss function. We show that Consistency Training increases generalization by a significant margin on all tested datasets when compared to conventional data augmentation. This framework is in other words a more performant alternative to data augmentation. Consistency Training leads to increased generalization with no additional overhead asides from the added computational cost involved with computing the auxiliary loss term and the memory required to store augmented and un-augmented versions of each batch. We summarize our contributions as following:

- We introduce Segmentation Inconsistency Loss (SIL), a novel region-based segmentation loss function which quantifies the inconsistency between two predicted segmentations when the inputs are subjected to arbitrary augmentations.
- We propose a robust method of incorporating this loss function without a loss of segmentation performance through a dynamic weighting method.
- We demonstrate quantitatively that Consistency Training increases generalization when compared to data augmentation on three OOD datasets.

## 2 Related Work

**Generalization Failure.** The development of consistency training was in large part informed by recent advances in the understanding of generalization failure. D'Amour et al. [9] perform a thorough analysis of generalization failure through multiple case studies and highlight the role of underspecification therein. Geirhos et al. [11] explore the idea of shortcut learning in a similar manner, and highlight the importance of learning causally related features. Schölkopf [26] discusses the importance of causality in machine learning and how it relates to generalization failure.

**Generalizable Training Methods.** Increasing generalizability is an open problem, and there exists a large diversity of different approaches and perspectives on the matter in the literature. Arjovsky et al. [4] develop a novel training paradigm that makes use of multiple training environments in order to increase generalization. Robey et al. [23] employ a similar method and develop a model-based training paradigm which attempts to induce invariance to learned mappings between training environments. Sandfort et al. [25] also leverage generative networks, but instead simply use generated

CT-images as data augmentation, which they show improves OOD performance. Gokhale et al. [12] compare the use of multiple data modification methods on robustness and generalization and find that data augmentation improves generalizability by a significant margin. Finally, Hendrycks et al. [16] incorporate a consistency term into their loss function, in particular Jensen-Shannon distance between output probabilities - in order to facilitate robustness to distributional shifts for the image-classification task.

**Generalizable Polyp Segmentation.** In the context of polyp-segmentation, this work was motivated in large part by the findings in the proceedings of EndoCV2021 [3], which through the evaluation of submissions on multiple OOD datasets highlighted the significance of generalization failure. The winning submission to EndoCV2021, submitted by Thambawita et al. [30], leverages an ensemble-network in order to increase generalizability. Honga et al. [17] also implement an ensemble-based model, which they show improves generalization. Gu et al. [14] make use of domain composition and attention in an attempt to generalize to unseen domains.

# 3 Approach

## 3.1 Consistency Training Method

This section will introduce Consistency Training, a training procedure wherein the objective is to optimize for invariance to a set of various image transformations by quantifying the degree to which the model outputs inconsistent predictions when its input is subjected to some transformations. This is achieved by giving the model two images: one which is augmented, and one which is not. These inputs are then passed through the model, resulting in two segmentation masks. The difference between these two predictions is then computed, and compared to the difference (if any) between the augmented and unaugmented segmentation labels. This is then incorporated into the loss-function such that the discrepancy between the expected prediction change and actual prediction change is minimized. This is illustrated in 1. The next sections will cover the theoretical basis of this training procedure as well as the implementation of its constituent components.

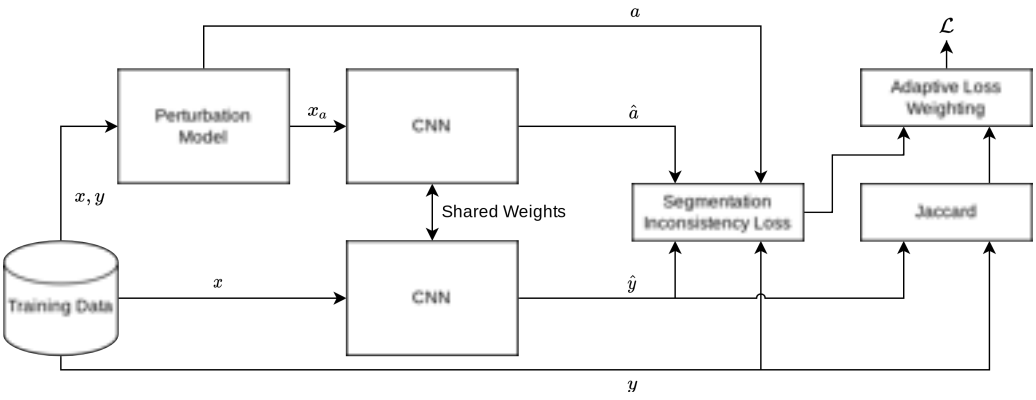


Figure 1: Diagram showing Consistency Training. The CNN is given two images, where one is simply an augmented version of the other. It then outputs two segmentations, which in conjunction with the labels for both images is used to compute SIL. The mIoU is then calculated, and used as weights for this term and a segmentation loss, in our case Jaccard loss.

## 3.2 Quantifying Segmentation Consistency

Let $Y := \{y, \hat{y} := f(x)\}$ be the set consisting of the segmentation labels (masks) and predictions for the unperturbed samples, where $f(\cdot)$ as before denotes the segmentation model. Let $\epsilon(\cdot)$ be some perturbation function. Then, let $A := \{a := \epsilon(y), \hat{a} := f(\epsilon(x))\}$ be the set consisting of masks and segmentation predictions when the input is subjected to a perturbation. Segmentation consistency can then be quantified as:

$$\mathcal{C}(y, a, \hat{y}, \hat{a}) = \frac{\sum\{y \cap a \cap \hat{y} \cap \hat{a}\}}{\sum\{y \cup a \cup \hat{y} \cup \hat{a}\}} \tag{1}$$

Equivalently, *inconsistency* can be quantified as:

$$\overline{\mathcal{C}}(y, a, \hat{y}, \hat{a}) = \frac{1}{\sum\{y \cup a \cup \hat{y} \cup \hat{a}\}} \sum\{y \ominus \hat{y} \ominus a \ominus \hat{a}\} \tag{2}$$

$\ominus$ here denotes the symmetric difference/disjunctive union. These formulations are, of course, related by:

$$\mathcal{C}(y, a, \hat{y}, \hat{a}) = 1 - \overline{\mathcal{C}}(y, a, \hat{y}, \hat{a})$$

In simple terms, this quantity corresponds to counting the number of pixels that change after the input is subjected to a perturbation,$\hat{a} \ominus \hat{y}$, but discounting those we expect to change, $a \ominus y$. This is shown in Figure 2.

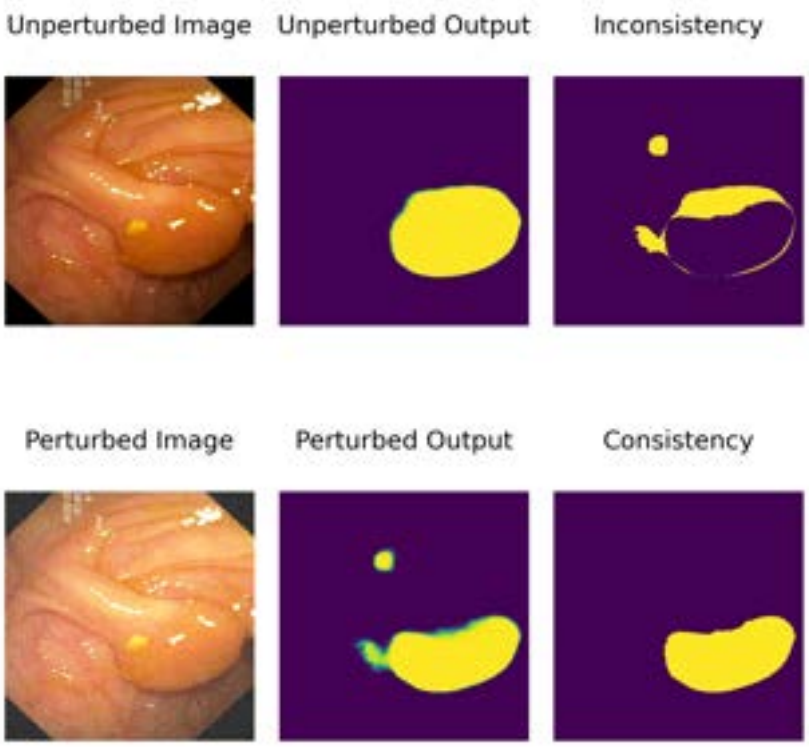


Figure 2: Examples of consistency and inconsistency calculation when the input is subjected to additive noise. The consistency for this sample is 0.68 and inconsistency 0.32, meaning that 64% of the pixels constitute consistent predictions across the two inputs.

Inconsistency as expressed in Equation (2) is not differentiable, and thus it cannot in its current state be used as a part of a loss function. Thus, a smooth extension of this metric is needed which can be

achieved in much the same way as how the Jaccard loss can be derived from the Jaccard index - i.e by using differentiable versions of the set functions.

We can extend the definition of the symmetric difference to $\Theta(A, B) = A(1 - B) + B(1 - A)$. This, naturally, is equivalent to the standard symmetric difference if the values of A and B are binary. Similarly, the union operator can be extended as $\bigcup(A, B) = A + B - AB$, and the intersection operator as $\bigcap(A, B) = AB$. Like its binary equivalents, these operators maintain their associative and commutative properties. One can optimize for consistency by replacing the operators in Equation (2) with these functions, which in turn can be used as a loss function:

$$L_c(y, \hat{y}, a, \hat{a}) = \sum \frac{\Theta(y, \hat{y}, a, \hat{a})}{\bigcup(y, \hat{y}, a, \hat{a})} \tag{3}$$

This loss function will from this point be referred to as the SIL.

### 3.3 Incorporating Consistency into Training

Using SIL as a loss function on its own is not really useful since it only expresses inconsistency, and is to a large extent agnostic to whatever object it is trying to segment. To illustrate, consider a model that predicts that every pixel is positive regardless of the content of the image, and that the augmentation strategy does not make use of augmentations that affect the labels. In this case, the consistency term will always be zero. For example, if the augmentation being performed is simply additive noise, the inconsistency term is equally well minimized if the model learns to predict that every pixel is positive as it would be if the model learned to be robust to additive noise. Consequently, it has to be combined with a segmentation loss, for instance Jaccard loss. A simple way to do this would be to simply add them together and normalize, i.e:

$$L(Y, A) = \frac{1}{2}\big[L_{seg}(Y) + L_c(Y, A)\big]$$

Preliminary experiments showed that this, however, exhibited some degree of instability during training. The model would readily get stuck in local minima where its predictions were indeed consistent, but also consistently predicting artifacts. Examples of this can be found in the Appendix.

To mitigate this, it is possible to employ a weighting strategy. Instead of simply adding the respective losses together, one may weight the individual components adaptively according to the InD segmentation performance, for instance Mean Intersection over Union (mIoU). This way, the model will learn to predict generally correct segmentations early in the training, then start weighting consistency and as a result generalization more and more as the model sees improvements to its segmentation performance:

$$L = (1 - IoU) \times L_{seg} + IoU \times L_c \tag{4}$$

Using this formulation, the model will start off trying to learn features that contribute to generally improved segmentation performance, then as segmentation performance improves start principally focusing on learning to be consistent. If the model starts veering into areas in the loss-landscape that constitute poor segmentation performance, it will self-correct by weighing the segmentation loss more. In the implementation used in this study, the mIoU weights were calculated on a per-batch basis such that the model can quickly adapt if either of the respective objectives exhibit a degradation in performance during training.

## 4 Experiments and Results

To determine the generalizability of our methods, we trained ten instances each of four separate models using Consistency Training, as as well as with conventional data augmentation and no augmentation, which served as baselines. The generalizability of these models was then determined through computing mIoU on three OOD datasets. The mIoUs across datasets for models trained with Consistency training was then compared to the mIoUs across datasets of the two baselines to ascertain the impact of Consistency Training relative to the baselines.

### 4.1 Experimental Setup

**Models.** To evaluate the impact of Consistency Training sufficiently, it was tested across a range of different models. These models include DeepLabV3+ [8], Feature Pyramid Network (FPN) [22], UNet [24], and Tri-Unet [30].

The models were implemented in pytorch using the segmentation-models-pytorch library [31], using the library's default values. This includes initialization with Imagenet-pretrained weights. Ten instances of each model were trained across each configuration in order to perform statistical analysis.

**Datasets.** The best way to evaluate the generalizability of a given predictor is to test it directly on OOD data. Though this can to some extent be achieved by carefully designing stress-tests [9], a more straight-forward approach is to simply leverage existing OOD datasets. To this end, a number of polyp-segmentation datasets were selected. The names, sizes, resolutions and availabilities of these datasets is shown in Table 1. Sample images and masks from the datasets can be seen in Figure 3. Kvasir-SEG was selected as the training dataset, and partitioned into a 80/10/10 split as training/validation/test data.

Table 1: Dataset Overview. The training dataset is marked using "*".

| Dataset | Resolution | Size | Availability |
|---|---|---|---|
| Kvasir-SEG* [20] | Variable | 1000 | Public |
| Etis-LaribDB [28] | 1255x966 | 196 | Public |
| CVC-ClinicDB [5] | 388x288 | 612 | Public |
| EndoCV2020 [2] | Variable | 127 | On Request |


Figure 3: Sample images from the datasets.

**Metrics** We used two metrics to evaluate generalizability. To evaluate raw performance, we used mIoU, which is defined as follows:

$$IoU(y, \hat{y}) = \frac{\sum\{y = \hat{y}\}}{\sum\{y = 1\} \cup \{\hat{y} = 1\}\}}$$

Measuring the average mIoU scores across all the aforementioned datasets, naturally, provides an indication of the generalizability of the given predictor. Though it is of course impossible to account for all distributional shifts that may occur in deployment, high degrees of generalization across multiple datasets should nevertheless indicate a sufficient level of generalization.

**Implementation details.** All experiments where conduced using Nvidia Tesla-V100 GPUs on the eX3 computing infrastructure offered by Simula Research Laboratory. The experiments were implemented in Python 3.8 using PyTorch 1.8.0 and segmentation-models-pytorch [31]. The source code as well as all of the raw data is available at `https://github.com/BirkTorpmannHagen/SegmentationConsistencyTraining`.

The augmentation method used both for the baseline and as part of Consistency Training was implemented using the albumentations library [7], and consisted of the following transformations: RandomRotate90, GaussNoise, ImageCompression, OpticalDistortion and ColorJitter. For the regular augmentation baseline, the augmentation probability was set to $0.5$, in which case all of the aforementioned transformations were applied. The hyperparameters used when training the models are shown in Table 2.

Table 2: Hyperparameters.

| Component | Type | Hyperparameters |
|---|---|---|
| Dataloader | - | $batch\_size = 8$<br>train/val/test split $= 80/10/10$ |
| Optimizer | Adam | $lr = 0.00001$ |
| Scheduler | Cosine Annealing w/ Warm Restarts | $T_0 = 50$<br>$T_{mult} = 2$ |
| Evaluation | Loss-based Early Stopping | $epochs = 300$ |

### 4.2 Out of Distribution Generalization

Table 3 shows the mean mIoUs for models trained with and without data augmentation, and models trained with Consistency Training. Comparing Consistency Training and conventional data augmentation for each model, statistical significance was achieved for all models except the TriUnet on the Etis-LaribDB dataset, for the FPN and Unet on the CVC-ClinicDB dataset, and for the Unet on the EndoCV2020 dataset after an independent-sample t-test. When averaging across models, Consistency Training improves generalization by a statistically significant margin ($p<0.01$) on all OOD datasets over conventional augmentation after a Mann-Whitney U-test. This is shown in Figure 4. This shows that Consistency Training can be considered a more generalizable alternative to data augmentation.

## 5 Discussion and Conclusion

In this paper, we introduced Segmentation Consistency Training, a novel training procedure for segmentation which explicitly optimizes for consistent behaviour when an input subjected to augmentation. We showed that this improves OOD generalization by a statistically significant amount across several models when compared to conventional data augmentation. Moreover, we show that Consistency Training mitigates underspecification to a greater extent than data augmentation by analyzing performance variability.

Table 3: Mean IoUs for training methods, precision truncated to 99% confidence. Consistency training entries with greater performance than conventional augmentation are highlighted in bold. If they are better by a statistically significant margin (p>0.99) after an independent sample two-sided t-test, they are also marked with a "*".

| Model | No Augmentation | Vanilla Augmentation | Consistency Training |
|---|---|---|---|
| **Kvasir-SEG (In-Distribution)** | | | |
| DD-DeepLabV3+ | 0.829 | 0.848 | **0.852** |
| DeepLab | 0.822 | 0.850 | **0.852** |
| FPN | 0.822 | **0.853** | 0.852 |
| TriUnet | 0.817 | 0.841 | **0.845** |
| Unet | 0.828 | **0.851** | **0.851** |
| **Etis-LaribDB (Out of Distribution)** | | | |
| DD-DeepLabV3+ | 0.408 | 0.460 | **0.482** |
| DeepLab | 0.417 | 0.472 | **0.505*** |
| FPN | 0.404 | 0.440 | **0.475*** |
| TriUnet | 0.309 | 0.410 | **0.434** |
| Unet | 0.403 | 0.447 | **0.481*** |
| **CVC-ClinicDB (Out of Distribution)** | | | |
| DD-DeepLabV3+ | 0.681 | 0.728 | **0.736** |
| DeepLabV3+ | 0.684 | 0.733 | **0.740** |
| FPN | 0.675 | 0.715 | **0.727*** |
| TriUnet | 0.623 | 0.684 | **0.696** |
| Unet | 0.679 | 0.717 | **0.730*** |
| **EndoCV2020 (Out of Distribution)** | | | |
| DD-DeepLabV3+ | 0.596 | **0.668** | **0.668** |
| DeepLab | 0.608 | **0.676** | **0.676** |
| FPN | 0.600 | 0.662 | **0.673** |
| TriUnet | 0.577 | 0.667 | **0.684** |
| Unet | 0.598 | 0.660 | **0.676*** |

### 5.1 Limitations

The batch size was kept constant across all experiments performed in this paper. However, as it can be argued that since Consistency Training implicitly increases the batch size, the experiments should ideally be repeated across a range of batch sizes.

Moreover, the experiments were only performed with one specific augmentation strategy. As it may be the case that the differences are less significant given a more highly developed augmentation strategy, repeating the experiment with a range of different augmentation strategies may be warranted.

As the experiments were only performed on polyp datasets, it can also be argued that it is uncertain whether Consistency Training has similar impacts on other segmentation tasks.

Finally, a larger number of samples should ideally have been collected across a wider diversity of model architectures. Increasing the granularity of the findings by other means, for instance by using a greater number of OOD datasets or designing parameterized stress-tests may also be warranted in order to develop a more thorough understanding of the impact of our methods.

### 5.2 Future Work

We plan to investigate a number of potential improvements of this framework. Consistency was for instance in this paper quantified as the symmetric difference between the expected change in the output due to augmentation and the actual change due to augmentation. This is largely agnostic to the augmentation being performed. However, it may be beneficial to take the nature of these augmentations into account. If the image is subjected to a 90 degree rotation, for instance, the

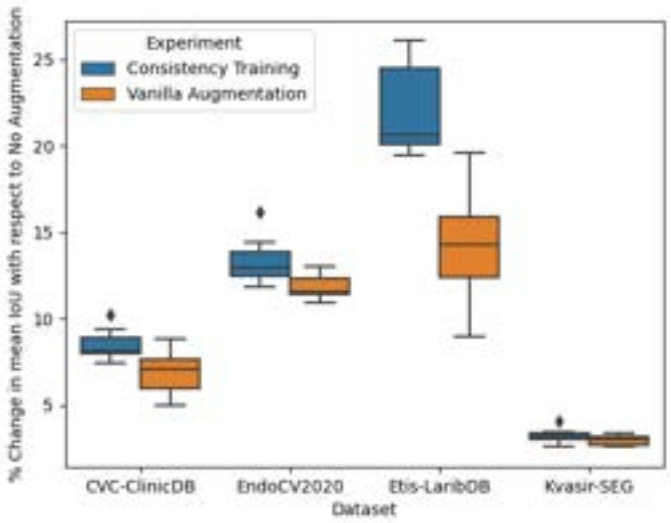


Figure 4: Improvements due Consistency Training and Data Augmentation as a percentage the mean mIoU without augmentation across datasets.

prediction would following the notion of consistency as used in this work be considered perfectly consistent so long as the pixels corresponding to the polyps are rotated, and the incorrectly classified pixels remain unchanged. However, if the model instead learns to rotate all of the pixels - even those that are incorrectly classified - it may learn a more accurate representation of what constitutes consistent behavior under rotation. I.e, instead of expressing inconsistency as in eq. (3), one can adjust the expected change term $a \oplus y$ to $\hat{y} \oplus \epsilon(\hat{y})$ such that also incorrect predictions can be considered consistent so long as they change in accordance to the nature of the perturbation model $\epsilon(\cdot)$. The resulting loss function can then be expressed as:

$$\bar{\mathcal{C}}(\hat{y}, \hat{a}) = \sum \frac{\Theta(\hat{y}, \hat{a}, \hat{y}, \epsilon(\hat{y}))}{\bigcup(\hat{y}, \hat{a}, \epsilon(\hat{y}))}$$

Which is equivalent to:

$$\bar{\mathcal{C}}(\hat{y}, \hat{a}) = \sum \frac{\Theta(\hat{y}, \hat{a}, \epsilon(\hat{y}))}{\bigcup(\hat{y}, \hat{a}, \epsilon(\hat{y}))}$$

This also has the advantage of being independent of the labels themselves. This may alleviate complications that may arise as a consequence of poor and/or incomplete labeling which would otherwise affect what the models learn to associate with consistent behaviour.

Repeating the experiments in this paper on a multitude of other segmentation tasks, for instance scene segmentation for autonomous vehicles, is also warranted. Evaluating Consistency Training through the use of stress-tests, for instance by augmenting datasets with a disjoint set of transformations as those used for training, may also provide some insights.

Further, one could investigate whether the consistency-training framework also can be implemented in the context of classification, object detection, or other applications of Deep Learning, and if similar improvements to generalizability can be shown in other domains.

Finally, one may compare the learned features of models trained with Consistency Training and the learned features of models trained conventionally. This could for instance be achieved through the use of Grad-CAM [27] or similar methods, and may be beneficial towards determining whether the model has learned at least partial invariance to the given augmentations.

## References


[1] Sharib Ali et al. *Assessing generalisability of deep learning-based polyp detection and segmentation methods through a computer vision challenge*. 2022. DOI: 10.48550/ARXIV.2202.12031. URL: https://arxiv.org/abs/2202.12031.

[2] Sharib Ali et al., eds. *EndoCV2020: 2nd International Workshop and Challenge on Computer Vision in Endoscopy*. Vol. 2595. Iowa, USA: CEUR Workshop Proceedings, 2020. URL: http://ceur-ws.org/Vol-2595/.

[3] Sharib Ali et al., eds. *Proceedings of the 3rd International Workshop and Challenge on Computer Vision in Endoscopy (EndoCV 2021) co-located with with the 17th IEEE International Symposium on Biomedical Imaging (ISBI 2021)*.

[4] Martin Arjovsky et al. *Invariant Risk Minimization*. 2019. DOI: 10.48550/ARXIV.1907.02893. URL: https://arxiv.org/abs/1907.02893.

[5] Jorge Bernal et al. "WM-DOVA maps for accurate polyp highlighting in colonoscopy: Validation vs. saliency maps from physicians". In: *Computerized Medical Imaging and Graphics* 43 (2015), pp. 99–111. DOI: http://dx.doi.org/10.1016/j.compmedimag.2015.02.007. URL: https://polyp.grand-challenge.org/CVCClinicDB/.

[6] Battista Biggio et al. "Evasion Attacks against Machine Learning at Test Time". In: *Lecture Notes in Computer Science* (2013), 387–402. ISSN: 1611-3349. DOI: 10.1007/978-3-642-40994-3_25. URL: http://dx.doi.org/10.1007/978-3-642-40994-3_25.

[7] A. Buslaev et al. "Albumentations: fast and flexible image augmentations". In: *ArXiv e-prints* (2018). eprint: 1809.06839. URL: https://albumentations.ai/.

[8] Liang-Chieh Chen et al. *Rethinking Atrous Convolution for Semantic Image Segmentation*. 2017. DOI: 10.48550/ARXIV.1706.05587. URL: https://arxiv.org/abs/1706.05587.

[9] Alexander D'Amour et al. *Underspecification Presents Challenges for Credibility in Modern Machine Learning*. 2020. DOI: 10.48550/ARXIV.2011.03395. URL: https://arxiv.org/abs/2011.03395.

[10] Logan Engstrom et al. *Exploring the Landscape of Spatial Robustness*. 2017. DOI: 10.48550/ARXIV.1712.02779. URL: https://arxiv.org/abs/1712.02779.

[11] Robert Geirhos et al. "Shortcut learning in deep neural networks". In: *Nature Machine Intelligence* 2.11 (Nov. 2020), 665–673. ISSN: 2522-5839. DOI: 10.1038/s42256-020-00257-z. URL: http://dx.doi.org/10.1038/s42256-020-00257-z.

[12] Tejas Gokhale et al. *Generalized but not Robust? Comparing the Effects of Data Modification Methods on Out-of-Domain Generalization and Adversarial Robustness*. 2022. DOI: 10.48550/ARXIV.2203.07653. URL: https://arxiv.org/abs/2203.07653.

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

[14] Ran Gu et al. "Domain Composition and Attention for Unseen-Domain Generalizable Medical Image Segmentation". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*. Ed. by Marleen de Bruijne et al. Cham: Springer International Publishing, 2021, pp. 241–250. ISBN: 978-3-030-87199-4. DOI: https://doi.org/10.1007/978-3-030-87199-4_23.

[15] Dan Hendrycks and Thomas Dietterich. *Benchmarking Neural Network Robustness to Common Corruptions and Perturbations*. 2019. arXiv: 1903.12261 [cs.LG].

[16] Dan Hendrycks et al. *AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty*. 2019. DOI: 10.48550/ARXIV.1912.02781. URL: https://arxiv.org/abs/1912.02781.

[17] Ayoung Honga et al. "Deep Learning Model Generalization with Ensemble in Endoscopic Images". In: *Proceedings of the 3rd International Workshop and Challenge on Computer Vision in Endoscopy (EndoCV 2021) co-located with with the 17th IEEE International Symposium on Biomedical Imaging (ISBI 2021)*. 2021, pp. 80–89. URL: http://ceur-ws.org/Vol-2886/paper8.pdf.

[18] Hossein Hosseini, Baicen Xiao, and Radha Poovendran. *Google's Cloud Vision API Is Not Robust To Noise*. 2017. arXiv: 1704.05051 [cs.CV].

[19] Andrew Ilyas et al. *Adversarial Examples Are Not Bugs, They Are Features*. 2019. DOI: 10.48550/ARXIV.1905.02175. URL: https://arxiv.org/abs/1905.02175.

[20] Debesh Jha et al. *Kvasir-SEG: A Segmented Polyp Dataset*. 2019. DOI: 10.48550/ARXIV.1911.07069. URL: https://arxiv.org/abs/1911.07069.

[21] Jacob Kauffmann et al. *The Clever Hans Effect in Anomaly Detection*. 2020. arXiv: 2006.10609 [cs.LG].

[22] Tsung-Yi Lin et al. "Feature Pyramid Networks for Object Detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[23] Alexander Robey, Hamed Hassani, and George J. Pappas. *Model-Based Robust Deep Learning: Generalizing to Natural, Out-of-Distribution Data*. 2020. arXiv: 2005.10247 [cs.LG].

[24] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. DOI: 10.48550/ARXIV.1505.04597. URL: https://arxiv.org/abs/1505.04597.

[25] Veit Sandfort et al. "Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks". In: *Scientific Reports* 9 (Nov. 2019). DOI: 10.1038/s41598-019-52737-x.

[26] Bernhard Schölkopf. *Causality for Machine Learning*. 2019. DOI: 10.48550/ARXIV.1911.10500. URL: https://arxiv.org/abs/1911.10500.

[27] Ramprasaath R. Selvaraju et al. "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization". In: *International Journal of Computer Vision* 128.2 (Oct. 2019), pp. 336–359. DOI: 10.1007/s11263-019-01228-7. URL: https://doi.org/10.1007%2Fs11263-019-01228-7.

[28] Juan Silva et al. "Toward embedded detection of polyps in wce images for early diagnosis of colorectal cancer". In: *International journal of computer assisted radiology and surgery* 9.2 (2014), pp. 283–293. DOI: https://doi.org/10.1007/s11548-013-0926-3.

[29] Vajira Thambawita et al. "An Extensive Study on Cross-Dataset Bias and Evaluation Metrics Interpretation for Machine Learning Applied to Gastrointestinal Tract Abnormality Classification". In: *ACM Trans. Comput. Healthcare* 1.3 (June 2020). ISSN: 2691-1957. DOI: 10.1145/3386295. URL: https://doi.org/10.1145/3386295.

[30] Vajira Thambawita et al. "DivergentNets: Medical Image Segmentation by Network Ensemble". In: *Proceedings of the 3rd International Workshop and Challenge on Computer Vision in Endoscopy (EndoCV 2021) co-located with with the 17th IEEE International Symposium on Biomedical Imaging (ISBI 2021)*. 2021, pp. 27–38. URL: https://arxiv.org/abs/2107.00283.

[31] Pavel Yakubovskiy. *Segmentation Models Pytorch*. https://github.com/qubvel/segmentation_models.pytorch. 2020.